

---

# **i.MX 6SoloLite Applications Processor Reference Manual**

Document Number: IMX6SLRM  
Rev. 1, 04/2013





## Contents

Section number	Title	Page
----------------	-------	------

### Chapter 1 Introduction

1.1	About This Document.....	125
1.1.1	Audience.....	125
1.1.2	Organization.....	125
1.1.3	Suggested Reading.....	126
1.1.3.1	General Information.....	126
1.1.3.2	Related Documentation.....	126
1.1.4	Conventions.....	126
1.1.5	Register Access.....	128
1.1.5.1	Register Diagram Field Access Type Legend.....	128
1.1.5.2	Register Macro Usage.....	128
1.1.6	Signal Conventions.....	129
1.1.7	Acronyms and Abbreviations.....	130
1.2	Introduction.....	131
1.3	Target Applications.....	132
1.4	Features.....	132
1.5	Architectural Overview.....	135
1.5.1	Simplified Block Diagram.....	135
1.5.2	Architectural Partitioning.....	136
1.5.3	Endianness Support.....	138
1.5.4	Memory Interfaces.....	138

### Chapter 2 Memory Maps

2.1	Memory system overview.....	139
2.2	ARM platform memory map.....	139
2.3	DMA memory map.....	145

Section number	Title	Page
<b>Chapter 3</b>		
<b>Interrupts and DMA Events</b>		
3.1	Overview.....	147
3.1.1	AP interrupts.....	147
3.2	SDMA event mapping.....	151
<b>Chapter 4</b>		
<b>External Signals and Pin Multiplexing</b>		
4.1	Overview.....	153
4.1.1	Pin Assignments.....	153
4.1.2	Muxing Options.....	208
<b>Chapter 5</b>		
<b>Fusemap</b>		
5.1	Boot Fusemap.....	255
5.2	Lock Fusemap.....	262
5.3	Fusemap Descriptions Table.....	263
<b>Chapter 6</b>		
<b>External Memory Controllers</b>		
6.1	Overview.....	271
6.2	Multi-mode DDR controller (MMDC) overview and feature summary.....	271
6.3	EIM-PSRAM/NOR Flash controller overview.....	272
6.3.1	EIM features.....	273
6.3.2	EIM boot scenarios.....	274
6.3.3	EIM boot configuration.....	274
6.3.4	OneNAND requirements.....	274
<b>Chapter 7</b>		
<b>System Debug</b>		
7.1	Overview.....	275
7.2	Chip and Cortex-A9 Core Platform Debug Architecture.....	275
7.2.1	Debug Features.....	276



Section number	Title	Page
7.2.2	Debug System components.....	277
7.2.2.1	AMBA trace bus (ATB).....	278
7.2.2.2	ATB replicator.....	278
7.2.2.3	Embedded Cross Triggering.....	278
7.2.2.3.1	Cross-Trigger Matrix (CTM).....	279
7.2.2.3.2	Cross-Trigger Interface (CTI).....	280
7.2.2.4	Debug Access Port (DAP).....	280
7.2.2.5	CoreSight trace port interface (TPIU).....	281
7.2.3	i.MX6SoloLite-Specific SJC Features.....	282
7.2.3.1	JTAG Disable Mode.....	282
7.2.3.2	JTAG ID.....	282
7.2.4	System JTAG Controller - SJC.....	283
7.2.5	System JTAG controller main features.....	283
7.2.6	SCJ TAP Port.....	283
7.2.7	SJC main blocks.....	283
7.3	Smart DMA (SDMA) core.....	284
7.3.1	SDMA On Chip Emulation Module (OnCE) Feature Summary.....	284
7.3.1.1	Other SDMA Debug Functionality.....	285
7.3.1.2	SDMA ROM Patching.....	286
7.4	Miscellaneous.....	286
7.4.1	Clock/Reset/Power.....	286
7.5	Supported tools.....	286

## Chapter 8 System Boot

8.1	Overview.....	289
8.2	Boot modes.....	290
8.2.1	Boot mode pin settings.....	290
8.2.2	High level boot sequence.....	291
8.2.3	Boot From Fuses Mode (BOOT_MODE[1:0] = 00b).....	292

Section number	Title	Page
8.2.4	Serial Downloader.....	293
8.2.5	Internal Boot Mode (BOOT_MODE[1:0] = 0b10).....	294
8.2.6	Boot security settings.....	295
8.3	Device Configuration.....	296
8.3.1	Boot eFUSE Descriptions.....	296
8.3.2	GPIO Boot Overrides.....	298
8.3.3	Device Configuration Data.....	300
8.4	Device Initialization.....	300
8.4.1	Internal ROM /RAM memory map.....	300
8.4.2	Boot Block Activation .....	301
8.4.3	Clocks at Boot Time.....	302
8.4.4	Enabling MMU and Caches.....	303
8.4.5	Exception Handling.....	304
8.4.6	Interrupt Handling During Boot.....	305
8.4.7	Persistent Bits.....	305
8.5	Boot Devices (Internal Boot).....	305
8.5.1	NOR Flash/OneNAND using EIM Interface.....	306
8.5.1.1	NOR Flash Boot Operation.....	306
8.5.1.2	OneNAND Flash Boot Operation.....	307
8.5.1.3	IOMUX Configuration for EIM Devices.....	308
8.5.2	Expansion Device.....	309
8.5.2.1	Expansion Device eFUSE Configuration.....	309
8.5.2.2	MMC and eMMC Boot.....	312
8.5.2.3	SD, eSD and SDXC.....	320
8.5.2.4	IOMUX Configuration for SD/MMC.....	320
8.5.2.5	Redundant Boot Support for Expansion Device.....	321
8.5.3	Serial ROM through SPI and I2C.....	322
8.5.3.1	Serial ROM eFUSE Configuration.....	323

Section number	Title	Page
8.5.3.2	I2C Boot.....	324
8.5.3.2.1	I2C IOMUX Pin Configuration.....	325
8.5.3.3	ECSPI Boot.....	325
8.5.3.3.1	ECSPI IOMUX Pin Configuration.....	327
8.6	Program image.....	328
8.6.1	Image Vector Table and Boot Data.....	328
8.6.1.1	Image Vector Table Structure.....	329
8.6.1.2	Boot Data Structure.....	330
8.6.2	Device Configuration Data (DCD).....	330
8.6.2.1	Write Data Command.....	331
8.6.2.2	Check Data Command.....	333
8.6.2.3	NOP Command.....	334
8.6.2.4	Unlock Command.....	335
8.7	Plugin Image.....	335
8.8	Serial Downloader.....	336
8.8.1	USB.....	337
8.8.1.1	USB Configuration Details.....	338
8.8.1.2	IOMUX Configuration for USB.....	338
8.8.2	Serial Download protocol.....	339
8.8.2.1	SDP Command.....	339
8.8.2.1.1	READ REGISTER.....	339
8.8.2.1.2	WRITE REGISTER.....	340
8.8.2.1.3	WRITE_FILE.....	341
8.8.2.1.4	ERROR_STATUS.....	342
8.8.2.1.5	DCD WRITE.....	343
8.8.2.1.6	JUMP ADDRESS.....	344
8.9	Recovery Devices.....	344
8.10	USB Low Power Boot.....	345
8.11	SD/MMC Manufacture Mode.....	346

Section number	Title	Page
8.12	High Assurance Boot (HAB).....	347
8.12.1	ROM Vector Table Addresses.....	348

## Chapter 9 Multimedia

9.1	Display and graphics subsystem.....	351
9.1.1	Electrophoretic Display Controller.....	352
9.1.2	SiPix Display Controller.....	353
9.1.3	PiXeL Pipeline.....	353
9.1.4	LCD Interface.....	353
9.1.5	CMOS Sensor Interface.....	354
9.1.6	2D Graphics Processing Unit (GPU2Dv2).....	354
9.1.6.1	2D feature summary.....	355
9.1.6.2	2D Performance.....	356
9.1.6.3	2D Software.....	356
9.1.7	Vector Graphics Processing Unit (GPUVGv2).....	356
9.1.7.1	Vector Graphics Features.....	356
9.1.7.2	Vector Graphics Performance.....	357
9.1.7.3	Vector Graphics Software.....	357
9.2	Audio subsystem.....	357
9.2.1	Audio subsystem module overview.....	357
9.2.2	Synchronous Serial Interface (SSI).....	359
9.2.3	Digital Audio MUX (AUDMUX).....	359
9.2.4	Sony/Philips Digital Interface (SPDIF).....	361

## Chapter 10 Clock and Power Management

10.1	Introduction.....	363
10.2	Device Power Management Architecture Components.....	363
10.2.1	Centralized components of clock generation and management.....	364
10.2.2	Centralized components of power generation, distribution and management.....	365

Section number	Title	Page
10.2.3	Reset generation and distribution system.....	365
10.2.4	Power and clock management framework.....	365
10.3	Clock Management.....	366
10.3.1	Centralized components of clock management system.....	366
10.3.2	Clock generation.....	369
10.3.2.1	Crystal Oscillator (XTALOSC) .....	369
10.3.2.2	LVDS I/O ports.....	369
10.3.2.3	PLLs.....	369
10.3.2.3.1	General PLL Control and Status Functions.....	371
10.3.2.4	CCM .....	372
10.3.2.5	Low Power Clock Gating unit (LPCG).....	373
10.3.3	Peripheral components of clock management system.....	373
10.3.3.1	Interface and functional clock.....	374
10.3.3.2	Block level clock management.....	374
10.3.3.2.1	Master clock protocol.....	375
10.3.3.2.2	Slave clock protocol.....	375
10.3.3.3	Clock Domain(s).....	376
10.3.3.4	Domain level clock management.....	376
10.3.3.5	Domain dependencies.....	376
10.4	Power management.....	376
10.4.1	Centralized Components of Power Management System.....	377
10.4.1.1	Integrated PMU.....	377
10.4.1.1.1	Digital LDO Regulators.....	378
10.4.1.1.2	Analog LDO regulators.....	379
10.4.1.1.3	USB LDO.....	380
10.4.1.1.4	SNVS regulator.....	380
10.4.1.1.5	Reverse well biasing.....	380
10.4.1.2	GPC - General Power Controller.....	380
10.4.1.3	SRC - System reset Controller.....	381

Section number	Title	Page
10.4.1.4	Power domain(s).....	382
10.4.1.4.1	Power distribution .....	382
10.4.1.4.2	Domain Memory and domain logic state retention in case of Power Gating...383	
10.4.1.4.3	Power Gating Domain Management.....	384
10.4.1.4.3.1	Cortex-A9 Core Platform.....	384
10.4.1.4.3.2	GPU2D.....	385
10.4.1.4.3.3	SoC.....	386
10.4.1.4.4	Power Gating domain dependencies.....	386
10.4.1.5	Voltage domains.....	387
10.4.1.6	Voltage domain management.....	387
10.4.1.6.1	Dynamic.....	387
10.4.1.6.1.1	DVFS.....	387
10.4.1.6.1.2	Voltage Scaling.....	388
10.4.1.6.2	Static .....	389
10.4.1.6.2.1	Standby Leakage reduction (SLR).....	389
10.4.1.6.3	Voltage domain dependencies.....	389
10.4.1.6.4	IO voltage .....	390
10.4.1.7	system domains layout.....	390
10.4.2	Power management techniques.....	392
10.4.2.1	Power saving techniques.....	393
10.4.2.2	Thermal-aware power management.....	393
10.4.2.3	Peripheral Power management.....	394
10.4.2.3.1	Main memory power management.....	394
10.4.2.3.2	Video-Graphics system power management.....	395
10.4.2.3.3	IO power reduction.....	395
10.4.3	Examples of External Power Supply Interfacing in the i.MX 6SoloLite based systems .....	395
10.5	ONOFF (Button).....	399

## Chapter 11

### System Security

11.1	Overview.....	401
11.2	Central Security Unit (CSU).....	403
11.2.1	CSU Overview.....	403
11.2.2	CSU Features.....	403
11.2.3	CSU Functional Description.....	404
11.2.3.1	CSU Peripheral Access Policy.....	404
11.3	Secure Non-Volatile Storage (SNVS).....	405
11.3.1	SNVS Overview.....	405
11.3.2	Tamper Detection.....	406
11.4	Data Co-Processor (DCP).....	406
11.5	High Assurance Boot (HAB).....	406
11.6	System JTAG Controller (SJC).....	407

## Chapter 12

### ARM Cortex A9 MPCore Platform (ARM)

12.1	Overview.....	409
12.2	External Signals.....	409
12.3	Platform configuration.....	411
12.3.1	Platform and SCU configuration.....	411
12.3.2	Core configuration.....	411
12.3.3	PL310 L2 Cache configuration.....	412
12.3.4	Endian Modes.....	412
12.3.5	Memory Parity error support .....	413
12.4	Performance and Power.....	413
12.4.1	Low-Power design.....	413
12.4.1.1	SRPG (State Retention Power Gating).....	413
12.4.1.2	Dynamic Voltage and Frequency Scaling (DVFS).....	414

Section number	Title	Page
12.4.2	Clocks, frequency goals.....	414
12.4.2.1	ARM Clock.....	414
12.4.2.2	Bus Clocks.....	414
12.4.2.3	Debug Clocks.....	414
12.5	Core Platform Sub-Blocks details.....	415
12.5.1	ARM Cortex A9 MPCore Processor.....	415
12.5.2	Media Processing Engine (MPE - NEON).....	415
12.5.3	Generic Interrupt Controller (GIC).....	416
12.5.3.1	Interrupt Controller Features.....	416
12.5.3.2	About the Interrupt Controller.....	416
12.5.3.3	Interrupt Controller Clock frequency.....	416
12.5.3.4	TrustZone support.....	416
12.5.4	Instruction and data caches (L1).....	417
12.5.4.1	L1 features.....	417
12.5.5	L2 Cache and controller (PL310).....	417
12.6	Debug and Trace Sub-blocks (CoreSight components).....	417
12.6.1	Debug Access Port (DAP) .....	418
12.6.2	Program Trace Macrocell (PTM).....	418
12.6.2.1	Program Flow Trace (PFT).....	419
12.6.3	Cross Trigger Interface (CTI).....	420
12.6.4	Embedded Trace Buffer (ETB).....	420
12.6.4.1	AMBA Trace Bus (ATB) Replicator .....	420

## Chapter 13

### AHB to IP Bridge (AIPSTZ)

13.1	Overview.....	421
13.1.1	Features.....	421
13.2	Clocks.....	421
13.3	General Operation.....	422
13.4	Functional Description.....	423



Section number	Title	Page
13.5	Access Protections.....	423
13.6	Access Support.....	423
13.7	Initialization Information.....	424
13.7.1	Security Block.....	424
13.8	AIPSTZ Memory Map/Register Definition.....	425
13.8.1	Master Priviledge Registers (AIPSTZ <sub>x</sub> _MPR).....	426
13.8.2	Off-Platform Peripheral Access Control Registers (AIPSTZ <sub>x</sub> _OPACR).....	428
13.8.3	Off-Platform Peripheral Access Control Registers (AIPSTZ <sub>x</sub> _OPACR1).....	431
13.8.4	Off-Platform Peripheral Access Control Registers (AIPSTZ <sub>x</sub> _OPACR2).....	434
13.8.5	Off-Platform Peripheral Access Control Registers (AIPSTZ <sub>x</sub> _OPACR3).....	437
13.8.6	Off-Platform Peripheral Access Control Registers (AIPSTZ <sub>x</sub> _OPACR4).....	440

## Chapter 14

### Digital Audio Multiplexer (AUDMUX)

14.1	Overview.....	443
14.1.1	Features.....	445
14.1.2	Modes and Operations.....	445
14.2	External Signals.....	445
14.3	Clocks.....	447
14.3.1	Clock Inputs.....	447
14.3.2	Clock Diagram.....	447
14.3.3	Clocking Restrictions.....	448
14.4	Default Register Configuration.....	448
14.4.1	Default Port Configuration.....	448
14.5	Functional Description.....	449
14.5.1	Operating Modes.....	449
14.5.1.1	Port Receive Data Modes.....	450
14.5.1.1.1	Normal Mode.....	451
14.5.1.1.2	Internal Network Mode.....	452
14.5.1.1.3	Transmit Data Output Enable Assertion.....	458

Section number	Title	Page
14.5.1.2	Tx/Rx Switch and External Network Mode.....	459
14.5.1.3	Timing Modes.....	460
14.5.1.3.1	Synchronous Mode (4-Wire Interface).....	460
14.5.1.3.2	Asynchronous Mode (6-Wire Interface).....	462
14.5.2	Connectivity Between Ports.....	465
14.5.2.1	Internal Port to External Port Connectivity.....	466
14.5.2.2	External Port to External Port Connectivity.....	467
14.5.2.3	Internal Port to Internal Port Connectivity.....	467
14.5.2.4	Loopback Connectivity.....	468
14.6	AUDMUX Memory Map/Register Definition.....	468
14.6.1	Port Timing Control Register 1 (AUDMUX_PTCR1).....	469
14.6.2	Port Data Control Register 1 (AUDMUX_PDCR1).....	471
14.6.3	Port Timing Control Register 2 (AUDMUX_PTCR2).....	472
14.6.4	Port Data Control Register 2 (AUDMUX_PDCR2).....	474
14.6.5	Port Timing Control Register 3 (AUDMUX_PTCR3).....	475
14.6.6	Port Data Control Register 3 (AUDMUX_PDCR3).....	477
14.6.7	Port Timing Control Register 4 (AUDMUX_PTCR4).....	478
14.6.8	Port Data Control Register 4 (AUDMUX_PDCR4).....	480
14.6.9	Port Timing Control Register 5 (AUDMUX_PTCR5).....	481
14.6.10	Port Data Control Register 5 (AUDMUX_PDCR5).....	483
14.6.11	Port Timing Control Register 6 (AUDMUX_PTCR6).....	484
14.6.12	Port Data Control Register 6 (AUDMUX_PDCR6).....	486
14.6.13	Port Timing Control Register 7 (AUDMUX_PTCR7).....	487
14.6.14	Port Data Control Register 7 (AUDMUX_PDCR7).....	489

## Chapter 15 Clock Controller Module (CCM)

15.1	Overview.....	491
15.1.1	Features.....	491
15.1.2	CCM Block Diagram.....	492

Section number	Title	Page
15.2	External Signals.....	494
15.3	CCM Clock Tree.....	494
15.4	System Clocks.....	496
15.5	Functional Description.....	500
15.5.1	Clock Generation.....	501
15.5.1.1	External Low Frequency Clock - CKIL .....	501
15.5.1.1.1	CKIL synchronizing to IPG_CLK.....	501
15.5.1.2	External High Frequency Clock - CKIH and internal oscillator.....	501
15.5.1.3	PLL reference clock.....	501
15.5.1.3.1	ARM PLL.....	502
15.5.1.3.2	USB PLLs.....	502
15.5.1.3.3	System PLL.....	502
15.5.1.3.4	Audio / Video PLL.....	502
15.5.1.3.5	Ethernet PLL.....	502
15.5.1.4	Phase Fractional Dividers (PFD).....	503
15.5.1.5	CCM internal clock generation.....	504
15.5.1.5.1	Clock Switcher.....	504
15.5.1.5.2	PLL bypass procedure.....	506
15.5.1.5.3	PLL clock change.....	506
15.5.1.5.4	Clock Root Generator.....	506
15.5.1.5.5	Initial values controlled by the System JTAG Controller (SJC).....	516
15.5.1.5.6	Divider change handshake.....	517
15.5.1.6	Disabling / Enabling PLLs.....	517
15.5.1.7	Low Power Clock Gating module (LPCG).....	517
15.5.1.7.1	MMDC handshake.....	519
15.5.2	DVFS support.....	520
15.5.3	Power modes.....	520
15.5.3.1	RUN mode.....	520

Section number	Title	Page
15.5.3.2	WAIT mode.....	520
15.5.3.2.1	Entering WAIT mode .....	520
15.5.3.2.2	Exiting WAIT mode .....	521
15.5.3.3	STOP mode.....	521
15.5.3.3.1	Entering STOP mode .....	521
15.5.3.3.2	Exiting STOP mode.....	522
15.6	CCM Memory Map/Register Definition.....	523
15.6.1	CCM Control Register (CCM_CCR).....	524
15.6.2	CCM Control Divider Register (CCM_CCDR).....	526
15.6.3	CCM Status Register (CCM_CSR).....	527
15.6.4	CCM Clock Swither Register (CCM_CCSR).....	528
15.6.5	CCM Arm Clock Root Register (CCM_CACRR).....	530
15.6.6	CCM Bus Clock Divider Register (CCM_CBCDR).....	531
15.6.7	CCM Bus Clock Multiplexer Register (CCM_CBCMR).....	533
15.6.8	CCM Serial Clock Multiplexer Register 1 (CCM_CSCMR1).....	536
15.6.9	CCM Serial Clock Multiplexer Register 2 (CCM_CSCMR2).....	539
15.6.10	CCM Serial Clock Divider Register 1 (CCM_CSCDR1).....	539
15.6.11	CCM SSI1 Clock Divider Register (CCM_CS1CDR).....	542
15.6.12	CCM SSI2 Clock Divider Register (CCM_CS2CDR).....	544
15.6.13	CCM D1 Clock Divider Register (CCM_CDCDR).....	545
15.6.14	CCM HSC Clock Divider Register (CCM_CHSCCDR).....	547
15.6.15	CCM Serial Clock Divider Register 2 (CCM_CSCDR2).....	548
15.6.16	CCM Serial Clock Divider Register 3 (CCM_CSCDR3).....	550
15.6.17	CCM Divider Handshake In-Process Register (CCM_CDHIPR).....	552
15.6.18	CCM Low Power Control Register (CCM_CLPCR).....	555
15.6.19	CCM Interrupt Status Register (CCM_CISR).....	558
15.6.20	CCM Interrupt Mask Register (CCM_CIMR).....	561
15.6.21	CCM Clock Output Source Register (CCM_CCOSR).....	564
15.6.22	CCM General Purpose Register (CCM_CGPR).....	566

Section number	Title	Page
15.6.23	CCM Clock Gating Register 0 (CCM_CCGR0).....	567
15.6.24	CCM Clock Gating Register 1 (CCM_CCGR1).....	569
15.6.25	CCM Clock Gating Register 2 (CCM_CCGR2).....	570
15.6.26	CCM Clock Gating Register 3 (CCM_CCGR3).....	572
15.6.27	CCM Clock Gating Register 4 (CCM_CCGR4).....	573
15.6.28	CCM Clock Gating Register 5 (CCM_CCGR5).....	574
15.6.29	CCM Clock Gating Register 6 (CCM_CCGR6).....	576
15.6.30	CCM Module Enable Override Register (CCM_CMEOR).....	577
15.7	CCM Analog Memory Map/Register Definition.....	578
15.7.1	Analog ARM PLL control Register (CCM_ANALOG_PLL_ARM $n$ ).....	581
15.7.2	Analog USB1 480MHz PLL Control Register (CCM_ANALOG_PLL_USB1 $n$ ).....	583
15.7.3	Analog USB2 480MHz PLL Control Register (CCM_ANALOG_PLL_USB2 $n$ ).....	585
15.7.4	Analog System PLL Control Register (CCM_ANALOG_PLL_SYS $n$ ).....	587
15.7.5	Analog Audio PLL control Register (CCM_ANALOG_PLL_AUDIO $n$ ).....	589
15.7.6	Numerator of Audio PLL Fractional Loop Divider Register (CCM_ANALOG_PLL_AUDIO_NUM)....	591
15.7.7	Denominator of Audio PLL Fractional Loop Divider Register (CCM_ANALOG_PLL_AUDIO_DENOM).....	592
15.7.8	Analog Video PLL control Register (CCM_ANALOG_PLL_VIDEO $n$ ).....	593
15.7.9	Numerator of Video PLL Fractional Loop Divider Register (CCM_ANALOG_PLL_VIDEO_NUM).....	595
15.7.10	Denominator of Video PLL Fractional Loop Divider Register (CCM_ANALOG_PLL_VIDEO_DENOM).....	596
15.7.11	Analog ENET PLL Control Register (CCM_ANALOG_PLL_ENET $n$ ).....	597
15.7.12	480MHz Clock (from PLL_USB2) Phase Fractional Divider Control Register (CCM_ANALOG_PFD_480 $n$ ).....	599
15.7.13	528MHz Clock (From PLL_SYS) Phase Fractional Divider Control Register (CCM_ANALOG_PFD_528 $n$ ).....	601
15.7.14	Miscellaneous Control Register (CCM_ANALOG_MISC0 $n$ ).....	603
15.7.15	Miscellaneous Control Register (CCM_ANALOG_MISC2 $n$ ).....	604

Section number	Title	Page
<b>Chapter 16</b>		
<b>CMOS Sensor Interface (CSI)</b>		
16.1	Overview.....	609
16.2	External Signals.....	610
16.3	Clocks.....	611
16.4	Principles of Operation.....	612
16.4.1	Data Transfer With The Embedded DMA Controllers.....	612
16.4.2	Gated Clock Mode.....	613
16.4.3	Non-Gated Clock Mode.....	614
16.4.4	CCIR656 Interlace Mode.....	614
16.4.5	CCIR656 Progressive Mode.....	616
16.4.6	Error Correction for CCIR656 Coding.....	617
16.5	Interrupt Generation.....	618
16.5.1	Start Of Frame Interrupt (SOF_INT).....	618
16.5.2	End Of Frame Interrupt (EOF_INT).....	618
16.5.3	Change Of Field Interrupt (COF_INT).....	618
16.5.4	CCIR Error Interrupt (ECC_INT).....	619
16.5.5	RxFIFO Full Interrupt (RxFF_INT).....	619
16.5.6	Statistic FIFO Full Interrupt (STATFF_INT).....	619
16.5.7	RxFIFO Overrun Interrupt (RFF_OR_INT).....	619
16.5.8	Statistic FIFO Overrun Interrupt (SFF_OR_INT).....	619
16.5.9	Frame Buffer1 DMA Transfer Done Interrupt (DMA_TSF_DONE_FB1).....	619
16.5.10	Frame Buffer2 DMA Transfer Done Interrupt (DMA_TSF_DONE_FB2).....	620
16.5.11	Statistic FIFO DMA Transfer Done Interrupt (DMA_TSF_DONE_SFF).....	620
16.5.12	AHB Bus Response Error Interrupt (HRESP_ERR_INT).....	620
16.6	Data Packing Style.....	620
16.6.1	RX FIFO Path.....	621
16.6.1.1	Bayer Data.....	621
16.6.1.2	RGB565 Data.....	621

Section number	Title	Page
16.6.1.3	RGB888 Data.....	622
16.6.2	STAT FIFO Path.....	624
16.7	CSI Memory Map/Register Definition.....	624
16.7.1	CSI Control Register 1 (CSI_CSICR1).....	625
16.7.2	CSI Control Register 2 (CSI_CSICR2).....	629
16.7.3	CSI Control Register 3 (CSI_CSICR3).....	631
16.7.4	CSI Statistic FIFO Register (CSI_CSISTATFIFO).....	633
16.7.5	CSI RX FIFO Register (CSI_CSIRFIFO).....	634
16.7.6	CSI RX Count Register (CSI_CSIRXCNT).....	634
16.7.7	CSI Status Register (CSI_CSISR).....	635
16.7.8	CSI DMA Start Address Register - for STATFIFO (CSI_CSIDMASA_STATFIFO).....	638
16.7.9	CSI DMA Transfer Size Register - for STATFIFO (CSI_CSIDMATS_STATFIFO).....	638
16.7.10	CSI DMA Start Address Register - for Frame Buffer1 (CSI_CSIDMASA_FB1).....	639
16.7.11	CSI DMA Transfer Size Register - for Frame Buffer2 (CSI_CSIDMASA_FB2).....	640
16.7.12	CSI Frame Buffer Parameter Register (CSI_CSIFBUF_PARA).....	640
16.7.13	CSI Image Parameter Register (CSI_CSIIMAG_PARA).....	641

## Chapter 17 Debug Monitor (DBGMON)

17.1	Overview.....	643
17.1.1	Features Summary.....	644
17.1.2	Functional Description.....	644
17.1.3	Application.....	644
17.2	DBGMON Memory Map/Register Definition.....	645
17.2.1	HW_DBGMON_CTRL (DBGMON_HW_DBGMON_CTRL).....	646
17.2.2	HW_DBGMON_MASTER_EN (DBGMON_HW_DBGMON_MASTER_EN).....	648
17.2.3	HW_DBGMON_IRQ (DBGMON_HW_DBGMON_IRQ).....	649
17.2.4	HW_DBGMON_TRAP_ADDR_LOW (DBGMON_HW_DBGMON_TRAP_ADDR_LOW).....	650
17.2.5	HW_DBGMON_TRAP_ADDR_HIGH (DBGMON_HW_DBGMON_TRAP_ADDR_HIGH).....	651
17.2.6	HW_DBGMON_TRAP_ID (DBGMON_HW_DBGMON_TRAP_ID).....	651

Section number	Title	Page
17.2.7	HW_DBGMON_SNVS_ADDR (DBGMON_HW_DBGMON_SNVS_ADDR).....	651
17.2.8	HW_DBGMON_SNVS_DATA (DBGMON_HW_DBGMON_SNVS_DATA).....	652
17.2.9	HW_DBGMON_SNVS_INFO (DBGMON_HW_DBGMON_SNVS_INFO).....	653
17.2.10	HW_DBGMON_VERSION (DBGMON_HW_DBGMON_VERSION).....	654

## Chapter 18 Data Co-Processor (DCP)

18.1	Overview.....	655
18.1.1	DCP Limitations for Software.....	657
18.2	Clocks.....	658
18.3	Operation.....	658
18.3.1	Memory Copy, Blit, and Fill Functionality.....	659
18.3.2	Advanced Encryption Standard (AES).....	659
18.3.2.1	Key Storage.....	659
18.3.2.2	AES OTP Key.....	660
18.3.2.3	Encryption Modes.....	660
18.3.3	Hashing.....	662
18.3.4	One Time Programmable (OTP) Key.....	663
18.3.5	Managing DCP Channel Arbitration and Performance.....	663
18.3.5.1	DCP Arbitration.....	664
18.3.5.2	Channel Recovery Timers.....	664
18.3.6	Programming Channel Operations.....	665
18.3.6.1	Virtual Channels.....	665
18.3.6.2	Context Switching.....	666
18.3.6.3	Working with Semaphores.....	667
18.3.6.4	Work Packet Structure.....	668
18.3.6.4.1	Next Command Address Field.....	668
18.3.6.4.2	Control0 Field.....	669
18.3.6.4.3	Control1 Field.....	671
18.3.6.4.4	Source Buffer.....	672



Section number	Title	Page
18.3.6.4.5	Destination Buffer.....	672
18.3.6.4.6	Buffer Size Field.....	673
18.3.6.4.7	Payload Pointer.....	673
18.3.6.4.8	Status.....	673
18.3.6.4.9	Payload.....	675
18.3.7	Programming DCP Functions.....	676
18.3.7.1	Basic Memory Copy Programming Example.....	676
18.3.7.2	Basic Hash Operation Programming Example.....	677
18.3.7.3	Basic Cipher Operation Programming Example.....	679
18.3.7.4	Multi-Buffer Scatter/Gather Cipher and Hash Operation Programming Example.....	680
18.4	DCP Memory Map/Register Definition.....	683
18.4.1	DCP Control Register 0 (DCP_CTRL).....	684
18.4.2	DCP Status Register (DCP_STAT).....	687
18.4.3	DCP Channel Control Register (DCP_CHANNELCTRL).....	689
18.4.4	DCP Capability 0 Register (DCP_CAPABILITY0).....	691
18.4.5	DCP Capability 1 Register (DCP_CAPABILITY1).....	692
18.4.6	DCP Context Buffer Pointer (DCP_CONTEXT).....	693
18.4.7	DCP Key Index (DCP_KEY).....	693
18.4.8	DCP Key Data (DCP_KEYDATA).....	694
18.4.9	DCP Work Packet 0 Status Register (DCP_PACKET0).....	695
18.4.10	DCP Work Packet 1 Status Register (DCP_PACKET1).....	695
18.4.11	DCP Work Packet 2 Status Register (DCP_PACKET2).....	699
18.4.12	DCP Work Packet 3 Status Register (DCP_PACKET3).....	700
18.4.13	DCP Work Packet 4 Status Register (DCP_PACKET4).....	700
18.4.14	DCP Work Packet 5 Status Register (DCP_PACKET5).....	701
18.4.15	DCP Work Packet 6 Status Register (DCP_PACKET6).....	701
18.4.16	DCP Channel 0 Command Pointer Address Register (DCP_CH0CMDPTR).....	702
18.4.17	DCP Channel 0 Semaphore Register (DCP_CH0SEMA).....	703
18.4.18	DCP Channel 0 Status Register (DCP_CH0STAT).....	704

Section number	Title	Page
18.4.19	DCP Channel 0 Options Register (DCP_CH0OPTS).....	706
18.4.20	DCP Channel 1 Command Pointer Address Register (DCP_CH1CMDPTR).....	707
18.4.21	DCP Channel 1 Semaphore Register (DCP_CH1SEMA).....	708
18.4.22	DCP Channel 1 Status Register (DCP_CH1STAT).....	709
18.4.23	DCP Channel 1 Options Register (DCP_CH1OPTS).....	711
18.4.24	DCP Channel 2 Command Pointer Address Register (DCP_CH2CMDPTR).....	712
18.4.25	DCP Channel 2 Semaphore Register (DCP_CH2SEMA).....	713
18.4.26	DCP Channel 2 Status Register (DCP_CH2STAT).....	714
18.4.27	DCP Channel 2 Options Register (DCP_CH2OPTS).....	716
18.4.28	DCP Channel 3 Command Pointer Address Register (DCP_CH3CMDPTR).....	717
18.4.29	DCP Channel 3 Semaphore Register (DCP_CH3SEMA).....	718
18.4.30	DCP Channel 3 Status Register (DCP_CH3STAT).....	719
18.4.31	DCP Channel 3 Options Register (DCP_CH3OPTS).....	721
18.4.32	DCP Debug Select Register (DCP_DBGSELECT).....	721
18.4.33	DCP Debug Data Register (DCP_DBGDATA).....	722
18.4.34	DCP Page Table Register (DCP_PAGETABLE).....	722
18.4.35	DCP Version Register (DCP_VERSION).....	723

## Chapter 19 Enhanced Configurable SPI (ECSPI)

19.1	Overview.....	725
19.1.1	Features.....	726
19.1.2	Modes and Operations.....	726
19.2	External Signals.....	727
19.3	Clocks.....	730
19.4	Functional Description.....	730
19.4.1	Master Mode.....	731
19.4.2	Slave Mode.....	731
19.4.3	Low Power Modes.....	732

Section number	Title	Page
19.4.4	Operations.....	732
19.4.4.1	Typical Master Mode.....	732
19.4.4.1.1	Master Mode with SPI_RDY.....	733
19.4.4.1.2	Master Mode with Wait States.....	735
19.4.4.1.3	Master Mode with SS_CTL[3:0] Control.....	735
19.4.4.1.4	Master Mode with Phase Control.....	736
19.4.4.2	Typical Slave Mode.....	737
19.4.5	Reset.....	738
19.4.6	Interrupts.....	738
19.4.7	DMA .....	739
19.4.8	Byte Order.....	740
19.5	Initialization.....	741
19.6	Applications.....	741
19.7	ECSPI Memory Map/Register Definition.....	742
19.7.1	Receive Data Register (ECSPIx_RXDATA).....	744
19.7.2	Transmit Data Register (ECSPIx_TXDATA).....	745
19.7.3	Control Register (ECSPIx_CONREG).....	745
19.7.4	Config Register (ECSPIx_CONFIGREG).....	748
19.7.5	Interrupt Control Register (ECSPIx_INTREG).....	750
19.7.6	DMA Control Register (ECSPIx_DMAREG).....	751
19.7.7	Status Register (ECSPIx_STATREG).....	753
19.7.8	Sample Period Control Register (ECSPIx_PERIODREG).....	754
19.7.9	Test Control Register (ECSPIx_TESTREG).....	756
19.7.10	Message Data Register (ECSPIx_MSGDATA).....	757

## Chapter 20

### External Interface Module (EIM)

20.1	Overview.....	759
20.1.1	Features.....	761

Section number	Title	Page
20.1.2	Modes of Operation.....	761
20.1.2.1	Asynchronous Mode.....	762
20.1.2.2	Asynchronous Page Read Mode.....	762
20.1.2.3	Multiplexed Address/Data Mode.....	762
20.1.2.4	Burst Clock Mode.....	763
20.1.2.5	Low Power Modes.....	764
20.1.2.6	Boot Mode.....	764
20.2	External Signals.....	764
20.2.1	Other Important Block I/O Signals Internal to the SoC.....	769
20.3	Clocks.....	770
20.4	Chip Select Memory Map.....	770
20.5	Functional Description.....	770
20.5.1	Continuous BCLK.....	771
20.5.2	Bus Sizing Configuration.....	772
20.5.2.1	8 BIT PORT SUPPORT.....	772
20.5.2.1.1	MOTOROLA 68000.....	772
20.5.2.1.2	INTEL 386.....	773
20.5.3	EIM Operational Modes.....	773
20.5.4	Burst Mode (Synchronous) Memory Operation.....	773
20.5.5	Burst Clock Divisor (BCD).....	774
20.5.6	Burst Clock Start (BCS).....	775
20.5.7	Multiplexed Address/Data Mode Support.....	775
20.5.8	Mixed Master/Memory Burst Modes Support.....	775
20.5.9	AXI (Master) Bus Cycles Support.....	776
20.5.10	WAIT_B Signal, RWSC and WWSC bit fields Usage.....	778
20.5.11	IPS Register Interface.....	778
20.5.12	MRS Set for PSRAM.....	779
20.5.13	EIM Access Termination .....	779
20.5.14	Error Conditions.....	779

Section number	Title	Page
20.5.15	DTACK Mode.....	780
20.5.16	RDY_INT Signal as Interrupt.....	780
20.5.17	RDY_INT Signal as Ready After Reset Indication.....	781
20.5.18	EIM_GRANT / EIM_BUSY Handshake Description.....	781
20.5.19	LPMD / LPACK Handshake Description.....	781
20.5.20	Endianness.....	782
20.5.21	Strobe Signal Use.....	783
20.6	Initialization Information.....	783
20.6.1	Booting from EIM.....	783
20.7	Typical Application.....	784
20.7.1	Access to AMD Flash.....	784
20.7.1.1	AMD Flash Asynchronous Mode Configuration.....	785
20.7.1.2	AMD Flash Utility.....	785
20.7.2	Access to Intel Sibley Flash.....	786
20.7.2.1	Intel Sibley Flash Asynchronous Mode Configuration.....	786
20.7.2.2	Intel Sibley Flash Synchronous Mode Configuration.....	786
20.7.2.3	Intel Sibley Flash Utility.....	786
20.7.3	Access to MDOC Device.....	787
20.7.3.1	MDOC Device Boot.....	787
20.7.3.2	MDOC Device Asynchronous Mode Configuration.....	787
20.7.3.3	MDOC Device Utility.....	787
20.7.4	Access to Micron PSRAM .....	787
20.7.4.1	Micron PSRAM Asynchronous Mode Configuration.....	788
20.7.4.2	Micron PSRAM Synchronous Mode Configuration.....	788
20.7.5	Access to Samsung OneNAND .....	788
20.7.5.1	Samsung OneNAND Boot.....	788
20.7.5.2	Samsung OneNAND Asynchronous Mode Configuration.....	789
20.7.5.3	Samsung OneNAND Synchronous Mode Configuration.....	789
20.7.5.4	Samsung OneNAND Utility.....	789

Section number	Title	Page
20.7.6	Access to Samsung UtRAM .....	790
20.7.6.1	Samsung UtRAM Asynchronous Mode Configuration.....	790
20.7.6.2	Samsung UtRAM Synchronous Mode Configuration.....	790
20.7.7	Access to Spansion Flash .....	790
20.7.7.1	Spansion Flash Asynchronous Mode Configuration.....	790
20.7.7.2	Spansion Flash Synchronous Mode Configuration.....	791
20.7.7.3	Spansion Flash Utility.....	791
20.7.8	8 bit support.....	792
20.8	External Bus Timing Diagrams.....	793
20.8.1	Asynchronous Read Memory Accesses Timing Diagram.....	793
20.8.2	Asynchronous Write Memory Accesses Timing Diagram.....	794
20.8.3	Asynchronous Read/Write Memory Accesses Timing Diagram.....	795
20.8.4	Asynchronous Read/Write Using RAL, WAL and CSREC.....	797
20.8.5	Consecutive Asynchronous Write Memory Accesses Timing Diagram.....	798
20.8.6	Consecutive Asynchronous Read Memory Accesses Timing Diagram.....	801
20.8.7	Burst (Synchronous Mode) Read Memory Accesses Timing Diagram - BCD=0.....	803
20.8.8	Burst (Synchronous Mode) Read Memory Accesses Timing Diagram - BCD=1.....	804
20.8.9	Asynchronous Page Mode Access.....	805
20.8.10	DTACK Mode - AXI Single Access.....	805
20.8.11	DTACK Mode - AXI Single Write Access.....	808
20.8.12	DTACK Mode - AXI Burst Access.....	809
20.9	EIM Memory Map/Register Definition.....	810
20.9.1	Chip Select n General Configuration Register 1 (EIM_CSnGCR1).....	813
20.9.2	Chip Select n General Configuration Register 2 (EIM_CSnGCR2).....	818
20.9.3	Chip Select n Read Configuration Register 1 (EIM_CSnRCR1).....	819
20.9.4	Chip Select n Read Configuration Register 2 (EIM_CSnRCR2).....	822
20.9.5	Chip Select n Write Configuration Register 1 (EIM_CSnWCR1).....	823
20.9.6	Chip Select n Write Configuration Register 2 (EIM_CSnWCR2).....	826
20.9.7	EIM Configuration Register (EIM_WCR).....	827

Section number	Title	Page
20.9.8	DLL Control Register (EIM_DCR).....	829
20.9.9	DLL Status Register (EIM_DSR).....	830
20.9.10	EIM IP Access Register (EIM_WIAR).....	831
20.9.11	Error Address Register (EIM_EAR).....	832

## Chapter 21 Enhanced LCD Interface (eLCDIF)

21.1	Overview.....	833
21.2	External Signals.....	833
21.3	Clocks.....	835
21.4	Functional Description.....	836
21.4.1	Bus Interface Mechanisms.....	837
21.4.1.1	Bus Master Operation in Write/Display Modes.....	838
21.4.1.2	System Bus Master Performance.....	838
21.4.2	Write Data Path.....	839
21.4.3	Read Data Path.....	845
21.4.4	eLCDIF Interrupts.....	850
21.4.5	Initializing the eLCDIF.....	850
21.4.5.1	Write Modes.....	850
21.4.5.2	MPU Read Mode.....	851
21.4.6	MPU Interface.....	852
21.4.6.1	Code Example to Initialize the eLCDIF in MPU Write Mode.....	854
21.4.7	VSYNC Interface.....	854
21.4.7.1	Code Example to Initialize eLCDIF in VSYNC Mode.....	855
21.4.8	DOTCLK Interface.....	856
21.4.8.1	Code Example.....	858
21.4.9	ITU-R BT.656 Digital Video Interface (DVI).....	858
21.4.10	eLCDIF Pin Usage by Interface Mode.....	859
21.5	Behavior During Reset.....	862

Section number	Title	Page
21.6	ELCDIF Memory Map/Register Definition.....	862
21.6.1	eLCDIF General Control Register (LCDIF_CTRL $n$ ).....	865
21.6.2	eLCDIF General Control1 Register (LCDIF_CTRL1 $n$ ).....	868
21.6.3	eLCDIF General Control2 Register (LCDIF_CTRL2 $n$ ).....	870
21.6.4	eLCDIF Horizontal and Vertical Valid Data Count Register (LCDIF_TRANSFER_COUNT).....	873
21.6.5	LCD Interface Current Buffer Address Register (LCDIF_CUR_BUF).....	873
21.6.6	LCD Interface Next Buffer Address Register (LCDIF_NEXT_BUF).....	874
21.6.7	LCD Interface Timing Register (LCDIF_TIMING).....	874
21.6.8	eLCDIF VSYNC Mode and Dotclk Mode Control Register0 (LCDIF_VDCTRL0 $n$ ).....	875
21.6.9	eLCDIF VSYNC Mode and Dotclk Mode Control Register1 (LCDIF_VDCTRL1).....	877
21.6.10	LCDIF VSYNC Mode and Dotclk Mode Control Register2 (LCDIF_VDCTRL2).....	877
21.6.11	eLCDIF VSYNC Mode and Dotclk Mode Control Register3 (LCDIF_VDCTRL3).....	878
21.6.12	eLCDIF VSYNC Mode and Dotclk Mode Control Register4 (LCDIF_VDCTRL4).....	879
21.6.13	Digital Video Interface Control0 Register (LCDIF_DVICTRL0).....	880
21.6.14	Digital Video Interface Control1 Register (LCDIF_DVICTRL1).....	880
21.6.15	Digital Video Interface Control2 Register (LCDIF_DVICTRL2).....	881
21.6.16	Digital Video Interface Control3 Register (LCDIF_DVICTRL3).....	882
21.6.17	Digital Video Interface Control4 Register (LCDIF_DVICTRL4).....	883
21.6.18	RGB to YCbCr 4:2:2 CSC Coefficient0 Register (LCDIF_CSC_COEFF0).....	884
21.6.19	RGB to YCbCr 4:2:2 CSC Coefficient1 Register (LCDIF_CSC_COEFF1).....	885
21.6.20	RGB to YCbCr 4:2:2 CSC Coefficient2 Register (LCDIF_CSC_COEFF2).....	886
21.6.21	RGB to YCbCr 4:2:2 CSC Coefficient3 Register (LCDIF_CSC_COEFF3).....	887
21.6.22	RGB to YCbCr 4:2:2 CSC Coefficient4 Register (LCDIF_CSC_COEFF4).....	887
21.6.23	RGB to YCbCr 4:2:2 CSC Offset Register (LCDIF_CSC_OFFSET).....	888
21.6.24	RGB to YCbCr 4:2:2 CSC Limit Register (LCDIF_CSC_LIMIT).....	889
21.6.25	LCD Interface Data Register (LCDIF_DATA).....	890
21.6.26	Bus Master Error Status Register (LCDIF_BM_ERROR_STAT).....	890
21.6.27	CRC Status Register (LCDIF_CRC_STAT).....	891
21.6.28	LCD Interface Status Register (LCDIF_STAT).....	891



Section number	Title	Page
21.6.29	LCD Interface Version Register (LCDIF_VERSION).....	893
21.6.30	LCD Interface Debug0 Register (LCDIF_DEBUG0).....	894
21.6.31	LCD Interface Debug1 Register (LCDIF_DEBUG1).....	896
21.6.32	LCD Interface Debug2 Register (LCDIF_DEBUG2).....	897
21.6.33	eLCDIF Threshold Register (LCDIF_THRES).....	897

## Chapter 22

### Electrophoretic Display Controller (EPDC)

22.1	Overview .....	899
22.1.1	EPDC Block Diagram.....	900
22.2	External Signals.....	901
22.3	Programming Model.....	903
22.3.1	Assumptions.....	904
22.3.2	Register Space (Write/Set/Clear/Toggle).....	904
22.3.3	Interrupts.....	905
22.3.3.1	Interrupt Sources.....	905
22.3.3.2	Enabling/Masking Interrupts.....	905
22.3.3.3	Handling/Clearing Interrupts.....	906
22.3.4	Controller Setup.....	906
22.3.4.1	Memory Requirements.....	906
22.3.4.2	Panel Architecture Configuration.....	907
22.3.4.3	TFT Panel Timing Configuration .....	910
22.3.4.4	Source Driver and Pixel Clock Configuration.....	916
22.3.4.5	Initializing the Display.....	918
22.3.4.5.1	Reset/Clocks and Buffer Preparation.....	918
22.3.4.5.2	Performing an Initialization Display Update.....	919
22.3.5	Dual-Scan Configuration.....	919
22.3.6	Update Buffer Analysis Functions.....	923
22.3.7	Waveform Mode Selection (AUTOWV).....	924
22.3.8	Panel Interface Generator (Pigeon Mode).....	925

Section number	Title	Page
22.3.9	Display Update Programming.....	928
22.3.9.1	Initiating a Display Update.....	928
22.3.9.2	Update Processing and Collisions.....	929
22.3.9.3	Multiple Update Flow.....	933
22.3.10	Architectural Clock Gating (Low Power Mode).....	936
22.3.11	Performance Tuning and Considerations.....	938
22.3.11.1	Memory and Bus Bandwidth Requirements.....	938
22.3.11.2	Pixel Latency FIFOs.....	939
22.3.11.3	Basic Watermarking Control.....	939
22.3.11.4	Update/Refresh Tuning (VSCAN_HOLD OFF).....	940
22.3.11.5	System-Level Arbitration Control.....	940
22.4	EPDC Memory Map/Register Definition.....	942
22.4.1	EPDC Control Register (EPDC_CTRLn).....	951
22.4.2	EPDC Waveform Address Pointer (EPDC_WVADDR).....	952
22.4.3	EPDC Working Buffer Address (EPDC_WB_ADDR).....	953
22.4.4	EPDC Screen Resolution (EPDC_RES).....	954
22.4.5	EPDC Format Control Register (EPDC_FORMATn).....	954
22.4.6	EPDC FIFO control register (EPDC_FIFOCTRLn).....	956
22.4.7	EPDC Update Region Address (EPDC_UPD_ADDR).....	957
22.4.8	EPDC Update Region Stride (EPDC_UPD_STRIDE).....	958
22.4.9	EPDC Update Command Co-ordinate (EPDC_UPD_CORD).....	958
22.4.10	EPDC Update Command Size (EPDC_UPD_SIZE).....	959
22.4.11	EPDC Update Command Control (EPDC_UPD_CTRLn).....	960
22.4.12	EPDC Update Fixed Pixel Control (EPDC_UPD_FIXEDn).....	961
22.4.13	EPDC Temperature Register (EPDC_TEMP).....	962
22.4.14	Waveform Mode Lookup Table Control Register. (EPDC_AUTOWV_LUT).....	962
22.4.15	EPDC Timing Control Engine Control Register (EPDC_TCE_CTRLn).....	963
22.4.16	EPDC Timing Control Engine Source-Driver Config Register (EPDC_TCE_SDCFGn).....	965
22.4.17	EPDC Timing Control Engine Gate-Driver Config Register (EPDC_TCE_GDCFGn).....	966

Section number	Title	Page
22.4.18	EPDC Timing Control Engine Horizontal Timing Register 1 (EPDC_TCE_HSCAN1n).....	967
22.4.19	EPDC Timing Control Engine Horizontal Timing Register 2 (EPDC_TCE_HSCAN2n).....	967
22.4.20	EPDC Timing Control Engine Vertical Timing Register (EPDC_TCE_VSCANn).....	968
22.4.21	EPDC Timing Control Engine OE timing control Register (EPDC_TCE_OEn).....	968
22.4.22	EPDC Timing Control Engine Driver Polarity Register (EPDC_TCE_POLARITYn).....	969
22.4.23	EPDC Timing Control Engine Timing Register 1 (EPDC_TCE_TIMING1n).....	970
22.4.24	EPDC Timing Control Engine Timing Register 2 (EPDC_TCE_TIMING2n).....	971
22.4.25	EPDC Timing Control Engine Timing Register 3 (EPDC_TCE_TIMING3n).....	972
22.4.26	EPDC Pigeon Mode Control Register 0 (EPDC_PIGEON_CTRL0n).....	972
22.4.27	EPDC Pigeon Mode Control Register 1 (EPDC_PIGEON_CTRL1n).....	973
22.4.28	EPDC IRQ Mask Register for LUT 0~31 (EPDC_IRQ_MASK1n).....	973
22.4.29	EPDC IRQ Mask Register for LUT 32~63 (EPDC_IRQ_MASK2n).....	974
22.4.30	EPDC Interrupt Register for LUT 0~31 (EPDC_IRQ1n).....	974
22.4.31	EPDC Interrupt Registerr for LUT 32~63 (EPDC_IRQ2n).....	975
22.4.32	EPDC IRQ Mask Register (EPDC_IRQ_MASKn).....	975
22.4.33	EPDC Interrupt Register (EPDC_IRQn).....	976
22.4.34	EPDC Status Register - LUTs (EPDC_STATUS_LUTS1n).....	977
22.4.35	EPDC Status Register - LUTs (EPDC_STATUS_LUTS2n).....	978
22.4.36	EPDC Status Register - Next Available LUT (EPDC_STATUS_NEXTLUT).....	978
22.4.37	EPDC LUT Collision Status (EPDC_STATUS_COL1n).....	980
22.4.38	EPDC LUT Collision Status (EPDC_STATUS_COL2n).....	981
22.4.39	EPDC General Status Register (EPDC_STATUSn).....	981
22.4.40	EPDC Collision Region Co-ordinate (EPDC_UPD_COL_CORD).....	983
22.4.41	EPDC Collision Region Size (EPDC_UPD_COL_SIZE).....	984
22.4.42	1-level Histogram Parameter Register. (EPDC_HIST1_PARAM).....	984
22.4.43	2-level Histogram Parameter Register. (EPDC_HIST2_PARAM).....	985
22.4.44	4-level Histogram Parameter Register. (EPDC_HIST4_PARAM).....	986
22.4.45	8-level Histogram Parameter 0 Register. (EPDC_HIST8_PARAM0).....	986
22.4.46	8-level Histogram Parameter 1 Register. (EPDC_HIST8_PARAM1).....	987

Section number	Title	Page
22.4.47	16-level Histogram Parameter 0 Register. (EPDC_HIST16_PARAM0).....	988
22.4.48	16-level Histogram Parameter Register. (EPDC_HIST16_PARAM1).....	989
22.4.49	16-level Histogram Parameter Register. (EPDC_HIST16_PARAM2).....	990
22.4.50	16-level Histogram Parameter Register. (EPDC_HIST16_PARAM3).....	990
22.4.51	EPDC General Purpose I/O Debug register (EPDC_GPION).....	991
22.4.52	EPDC Version Register (EPDC_VERSION).....	992
22.4.53	Panel Interface Signal Generator Register 0_0 (EPDC_PIGEON_0_0).....	993
22.4.54	Panel Interface Signal Generator Register 0_1 (EPDC_PIGEON_0_1).....	994
22.4.55	Panel Interface Signal Generator Register 0_1 (EPDC_PIGEON_0_2).....	995
22.4.56	Panel Interface Signal Generator Register 1_0 (EPDC_PIGEON_1_0).....	995
22.4.57	Panel Interface Signal Generator Register 1_1 (EPDC_PIGEON_1_1).....	996
22.4.58	Panel Interface Signal Generator Register 1_1 (EPDC_PIGEON_1_2).....	997
22.4.59	Panel Interface Signal Generator Register 2_0 (EPDC_PIGEON_2_0).....	998
22.4.60	Panel Interface Signal Generator Register 2_1 (EPDC_PIGEON_2_1).....	999
22.4.61	Panel Interface Signal Generator Register 2_1 (EPDC_PIGEON_2_2).....	999
22.4.62	Panel Interface Signal Generator Register 3_0 (EPDC_PIGEON_3_0).....	1000
22.4.63	Panel Interface Signal Generator Register 3_1 (EPDC_PIGEON_3_1).....	1001
22.4.64	Panel Interface Signal Generator Register 3_1 (EPDC_PIGEON_3_2).....	1002
22.4.65	Panel Interface Signal Generator Register 4_0 (EPDC_PIGEON_4_0).....	1002
22.4.66	Panel Interface Signal Generator Register 4_1 (EPDC_PIGEON_4_1).....	1004
22.4.67	Panel Interface Signal Generator Register 4_1 (EPDC_PIGEON_4_2).....	1004
22.4.68	Panel Interface Signal Generator Register 5_0 (EPDC_PIGEON_5_0).....	1005
22.4.69	Panel Interface Signal Generator Register 5_1 (EPDC_PIGEON_5_1).....	1006
22.4.70	Panel Interface Signal Generator Register 5_1 (EPDC_PIGEON_5_2).....	1007
22.4.71	Panel Interface Signal Generator Register 6_0 (EPDC_PIGEON_6_0).....	1007
22.4.72	Panel Interface Signal Generator Register 6_1 (EPDC_PIGEON_6_1).....	1008
22.4.73	Panel Interface Signal Generator Register 6_1 (EPDC_PIGEON_6_2).....	1009
22.4.74	Panel Interface Signal Generator Register 7_0 (EPDC_PIGEON_7_0).....	1010
22.4.75	Panel Interface Signal Generator Register 7_1 (EPDC_PIGEON_7_1).....	1011

Section number	Title	Page
22.4.76	Panel Interface Signal Generator Register 7_1 (EPDC_PIGEON_7_2).....	1011
22.4.77	Panel Interface Signal Generator Register 8_0 (EPDC_PIGEON_8_0).....	1012
22.4.78	Panel Interface Signal Generator Register 8_1 (EPDC_PIGEON_8_1).....	1013
22.4.79	Panel Interface Signal Generator Register 8_1 (EPDC_PIGEON_8_2).....	1014
22.4.80	Panel Interface Signal Generator Register 9_0 (EPDC_PIGEON_9_0).....	1014
22.4.81	Panel Interface Signal Generator Register 9_1 (EPDC_PIGEON_9_1).....	1016
22.4.82	Panel Interface Signal Generator Register 9_1 (EPDC_PIGEON_9_2).....	1016
22.4.83	Panel Interface Signal Generator Register 10_0 (EPDC_PIGEON_10_0).....	1017
22.4.84	Panel Interface Signal Generator Register 10_1 (EPDC_PIGEON_10_1).....	1018
22.4.85	Panel Interface Signal Generator Register 10_1 (EPDC_PIGEON_10_2).....	1019
22.4.86	Panel Interface Signal Generator Register 11_0 (EPDC_PIGEON_11_0).....	1019
22.4.87	Panel Interface Signal Generator Register 11_1 (EPDC_PIGEON_11_1).....	1020
22.4.88	Panel Interface Signal Generator Register 11_1 (EPDC_PIGEON_11_2).....	1021
22.4.89	Panel Interface Signal Generator Register 12_0 (EPDC_PIGEON_12_0).....	1022
22.4.90	Panel Interface Signal Generator Register 12_1 (EPDC_PIGEON_12_1).....	1023
22.4.91	Panel Interface Signal Generator Register 12_1 (EPDC_PIGEON_12_2).....	1023
22.4.92	Panel Interface Signal Generator Register 13_0 (EPDC_PIGEON_13_0).....	1024
22.4.93	Panel Interface Signal Generator Register 13_1 (EPDC_PIGEON_13_1).....	1025
22.4.94	Panel Interface Signal Generator Register 13_1 (EPDC_PIGEON_13_2).....	1026
22.4.95	Panel Interface Signal Generator Register 14_0 (EPDC_PIGEON_14_0).....	1026
22.4.96	Panel Interface Signal Generator Register 14_1 (EPDC_PIGEON_14_1).....	1028
22.4.97	Panel Interface Signal Generator Register 14_1 (EPDC_PIGEON_14_2).....	1028
22.4.98	Panel Interface Signal Generator Register 15_0 (EPDC_PIGEON_15_0).....	1029
22.4.99	Panel Interface Signal Generator Register 15_1 (EPDC_PIGEON_15_1).....	1030
22.4.100	Panel Interface Signal Generator Register 15_1 (EPDC_PIGEON_15_2).....	1031
22.4.101	EPDC Working Buffer Address for TCE (EPDC_WB_ADDR_TCE).....	1031

Section number	Title	Page
<b>Chapter 23</b>		
<b>Enhanced Periodic Interrupt Timer (EPIT)</b>		
23.1	Overview.....	1033
23.1.1	EPIT features.....	1033
23.1.2	EPIT modes and operations.....	1034
23.2	External signals.....	1034
23.3	Clocks.....	1034
23.4	Functional Description.....	1036
23.4.1	Operating modes.....	1036
23.4.1.1	Operating in set-and-forget mode.....	1036
23.4.1.2	Operating in free-running mode.....	1036
23.4.2	Operations.....	1037
23.4.3	Compare Event.....	1037
23.4.3.1	Counter Value Overwrite.....	1038
23.4.3.2	Low-Power Mode Behavior.....	1039
23.4.3.3	Debug Mode Behavior.....	1039
23.5	Initialization/ Application Information.....	1039
23.5.1	Change of Clock Source.....	1039
23.6	EPIT Memory Map/Register Definition.....	1040
23.6.1	Control register (EPIT <sub>x</sub> _CR).....	1040
23.6.2	Status register (EPIT <sub>x</sub> _SR).....	1043
23.6.3	Load register (EPIT <sub>x</sub> _LR).....	1044
23.6.4	Compare register (EPIT <sub>x</sub> _CMPR).....	1044
23.6.5	Counter register (EPIT <sub>x</sub> _CNR).....	1045
<b>Chapter 24</b>		
<b>Fast Ethernet Controller (FEC)</b>		
24.1	Overview.....	1047
24.1.1	Features.....	1049
24.2	External Signals.....	1050

Section number	Title	Page
24.3	Clocks.....	1050
24.4	Modes of Operation.....	1050
24.4.1	Full- and Half-Duplex Operation.....	1050
24.4.2	Interface Options.....	1051
24.4.2.1	10 Mbps and 100 Mbps RMII Interface.....	1051
24.4.2.2	10-Mbps 7-Wire Interface Operation.....	1051
24.4.3	Address Recognition Options.....	1051
24.4.4	Internal Loopback.....	1052
24.5	Functional Description.....	1052
24.5.1	Network Interface Options.....	1052
24.5.2	FEC Frame Transmission.....	1053
24.5.2.1	Transmit Inter-Packet Gap (IPG) Time.....	1054
24.5.2.2	Collision Handling.....	1054
24.5.2.3	Transmission Error Handling.....	1055
24.5.2.3.1	Transmitter Underrun .....	1055
24.5.2.3.2	Retransmission Attempts Limit Expired .....	1055
24.5.2.3.3	Late Collision .....	1056
24.5.2.3.4	Heartbeat .....	1056
24.5.3	FEC Frame Reception.....	1056
24.5.3.1	Receive Inter-Packet Gap (IPG) Time.....	1057
24.5.3.2	Ethernet Address Recognition.....	1057
24.5.3.2.1	Hash Algorithm.....	1060
24.5.3.3	Reception Error Handling.....	1063
24.5.3.3.1	Overflow .....	1063
24.5.3.3.2	Non-Octet (Dribbling Bits) .....	1063
24.5.3.3.3	CRC .....	1063
24.5.3.3.4	Frame Length Violation.....	1064
24.5.3.3.5	Truncation.....	1064
24.5.4	Full-Duplex Flow Control.....	1064

Section number	Title	Page
24.5.5	Internal and External Loopback.....	1065
24.6	Initialization/Application Information.....	1066
24.6.1	Initialization Sequence.....	1066
24.6.1.1	Hardware Controlled Initialization.....	1066
24.6.1.2	User Initialization (Prior to Asserting FEC_ECR[ETHER_EN]).....	1067
24.6.1.3	Microcontroller Initialization.....	1068
24.6.1.4	User Initialization (after asserting FEC_ECR[ETHER_EN]).....	1068
24.6.2	Buffer Descriptors.....	1068
24.6.2.1	Driver/DMA Operation with Buffer Descriptors.....	1069
24.6.2.2	Ethernet Transmit Buffer Descriptor (TxBD).....	1069
24.6.2.2.1	Driver/DMA Operation with Transmit Buffer Descriptors.....	1071
24.6.2.2.1.1	Transmit Frame in Multiple Buffers.....	1071
24.6.2.3	Ethernet Receive Buffer Descriptor (RxBD).....	1072
24.6.2.3.1	Driver/DMA Operation with Receive Buffer Descriptors.....	1073
24.7	Programmable Registers.....	1074
24.7.1	Top Level Block Memory Map.....	1074
24.7.2	Message Information Block (MIB) Counters Memory Map.....	1075
24.7.3	MIIGSK Registers Memory Map.....	1077
24.8	FEC Memory Map/Register Definition.....	1077
24.8.1	Ethernet interrupt event register (FEC_EIR).....	1079
24.8.2	Ethernet interrupt mask register (FEC_EIMR).....	1081
24.8.3	Receive descriptor active register (FEC_RDAR).....	1084
24.8.4	Transmit descriptor active register (FEC_TDAR).....	1085
24.8.5	Ethernet control register (FEC_ECR).....	1086
24.8.6	MII management frame register (FEC_MMFR).....	1087
24.8.7	MII speed control register (FEC_MSCR).....	1088
24.8.8	MIB control register (FEC_MIBC).....	1090
24.8.9	Receive control register (FEC_RCR).....	1091
24.8.10	Transmit control register (FEC_TCR).....	1093



Section number	Title	Page
24.8.11	Physical address low register (FEC_PALR).....	1095
24.8.12	Physical address upper register (FEC_PAUR).....	1095
24.8.13	Opcode and pause duration register (FEC_OPDR).....	1096
24.8.14	Descriptor individual address upper register (FEC_IAUR).....	1097
24.8.15	Descriptor individual address lower register (FEC_IALR).....	1097
24.8.16	Descriptor group address upper register (FEC_GAUR).....	1098
24.8.17	Descriptor group address lower register (FEC_GALR).....	1098
24.8.18	Transmit FIFO watermark register (FEC_TFWR).....	1099
24.8.19	FIFO receive bound register (FEC_FRBR).....	1100
24.8.20	FIFO receive FIFO start registers (FEC_FRSR).....	1100
24.8.21	Receive buffer descriptor ring start register (FEC_ERDSR).....	1101
24.8.22	Transmit buffer descriptor ring start register (FEC_ETDSR).....	1102
24.8.23	Maximum receive buffer size register (FEC_EMRBR).....	1102

## Chapter 25

### General Power Controller (GPC)

25.1	Overview.....	1105
25.2	Clocks.....	1106
25.3	DVFS overview.....	1106
25.3.1	Features.....	1108
25.4	Component Blocks description.....	1109
25.4.1	dvfs_stdb_smpl block.....	1109
25.4.2	dvfs_sig_wt block.....	1109
25.4.3	dvfs_pre_avg block.....	1110
25.4.4	dvfs_ld_add block.....	1112
25.4.5	dvfs_ema_avg block.....	1113
25.4.6	dvfs_thres_cmp block.....	1117
25.4.7	dvfs_thresh_count block.....	1118
25.4.8	Load Tracking Buffer Register.....	1119
25.4.9	Frequency Pattern Generator.....	1119

Section number	Title	Page
25.5	DVFS output event/interrupt configuration.....	1121
25.5.1	Interrupts.....	1121
25.5.2	DVFS Change Request Sequence Diagrams.....	1121
25.6	Power Gating Control (PGC).....	1122
25.6.1	Overview.....	1123
25.6.1.1	Features.....	1124
25.7	GPC Interrupt Controller (INTC).....	1124
25.7.1	Interrupt Controller features.....	1124
25.8	GPC Memory Map/Register Definition.....	1125
25.8.1	GPC Interface control register (GPC_CNTR).....	1125
25.8.2	GPC Power Gating Register (GPC_PGR).....	1128
25.8.3	IRQ masking register 1 (GPC_IMR1).....	1128
25.8.4	IRQ masking register 2 (GPC_IMR2).....	1129
25.8.5	IRQ masking register 3 (GPC_IMR3).....	1129
25.8.6	IRQ masking register 4 (GPC_IMR4).....	1130
25.8.7	IRQ status resister 1 (GPC_ISR1).....	1130
25.8.8	IRQ status resister 2 (GPC_ISR2).....	1131
25.8.9	IRQ status resister 3 (GPC_ISR3).....	1131
25.8.10	IRQ status resister 4 (GPC_ISR4).....	1132
25.9	PGC Memory Map/Register Definition.....	1132
25.9.1	PGC Control Register (PGC_DISPLAY_CTRL).....	1133
25.9.2	Power Up Sequence Control Register (PGC_DISPLAY_PUPSCR).....	1134
25.9.3	Pull Down Sequence Control Register (PGC_DISPLAY_PDNSCR).....	1135
25.9.4	Power Gating Controller Status Register (PGC_DISPLAY_SR).....	1135
25.9.5	PGC Control Register (PGC_GPU_CTRL).....	1136
25.9.6	Power Up Sequence Control Register (PGC_GPU_PUPSCR).....	1137
25.9.7	Pull Down Sequence Control Register (PGC_GPU_PDNSCR).....	1137
25.9.8	Power Gating Controller Status Register (PGC_GPU_SR).....	1138
25.9.9	PGC Control Register (PGC_CPU_CTRL).....	1139

Section number	Title	Page
25.9.10	Power Up Sequence Control Register (PGC_CPU_PUPSCR).....	1139
25.9.11	Pull Down Sequence Control Register (PGC_CPU_PDNSCR).....	1140
25.9.12	Power Gating Controller Status Register (PGC_CPU_SR).....	1141
25.10	DVFS Memory Map/Register Definition.....	1141
25.10.1	DVFS Thresholds (DVFS_THRS).....	1142
25.10.2	DVFS Counters thresholds (DVFS_COUN).....	1143
25.10.3	DVFS general purpose bits weight (DVFS_SIG1).....	1143
25.10.4	DVFS general purpose bits weight (DVFS_DVFS_SIG0).....	1144
25.10.5	DVFS general purpose bit 0 weight counter (DVFS_DVFSGPC0).....	1145
25.10.6	DVFS general purpose bit 1 weight counter (DVFS_DVFSGPC1).....	1146
25.10.7	DVFS general purpose bits enables (DVFS_DVFSGPBT).....	1147
25.10.8	DVFS EMAC settings (DVFS_DVFS_MAC).....	1149
25.10.9	DVFS Control (DVFS_CNTR).....	1150
25.10.10	DVFS Load Tracking Register 0, portion 0 (DVFS_DVFSLTR0_0).....	1153
25.10.11	DVFS Load Tracking Register 0, portion 1 (DVFS_DVFSLTR0_1).....	1154
25.10.12	DVFS Load Tracking Register 1, portion 0 (DVFS_DVFSLTR1_0).....	1155
25.10.13	DVFS Load Tracking Register 3, portion 1 (DVFS_DVFSLTR1_1).....	1155
25.10.14	DVFS pattern 0 length (DVFS_DVFSPT0).....	1156
25.10.15	DVFS pattern 1 length (DVFS_DVFSPT1).....	1157
25.10.16	DVFS pattern 2 length (DVFS_DVFSPT2).....	1157
25.10.17	DVFS pattern 3 length (DVFS_DVFSPT3).....	1158

## Chapter 26

### General Purpose Input/Output (GPIO)

26.1	Overview.....	1159
26.1.1	Block Diagram.....	1161
26.1.2	Features.....	1162
26.2	External Signals.....	1163
26.3	Clocks.....	1167

Section number	Title	Page
26.4	GPIO Functional Description.....	1167
26.4.1	GPIO Function.....	1167
26.4.2	GPIO Programming.....	1168
26.4.2.1	GPIO Read Mode.....	1168
26.4.2.2	GPIO Write Mode.....	1168
26.4.3	Interrupt Control Unit.....	1169
26.5	GPIO Memory Map/Register Definition.....	1169
26.5.1	GPIO data register (GPIOx_DR).....	1171
26.5.2	GPIO direction register (GPIOx_GDIR).....	1172
26.5.3	GPIO pad status register (GPIOx_PSR).....	1172
26.5.4	GPIO interrupt configuration register1 (GPIOx_ICR1).....	1173
26.5.5	GPIO interrupt configuration register2 (GPIOx_ICR2).....	1177
26.5.6	GPIO interrupt mask register (GPIOx_IMR).....	1180
26.5.7	GPIO interrupt status register (GPIOx_ISR).....	1181
26.5.8	GPIO edge select register (GPIOx_EDGE_SEL).....	1182

## Chapter 27 General Purpose Timer (GPT)

27.1	Overview.....	1183
27.1.1	Features.....	1185
27.1.2	Modes and Operation.....	1185
27.2	External Signals.....	1185
27.2.1	External Clock Input .....	1186
27.2.2	Input Capture Trigger Signals .....	1186
27.2.3	Output Compare Signals.....	1187
27.3	Clocks.....	1187
27.4	Functional Description.....	1189
27.4.1	Operating Modes.....	1189
27.4.1.1	Restart Mode.....	1189
27.4.1.2	Free-Run Mode.....	1189

Section number	Title	Page
27.4.2	Operation.....	1190
27.4.2.1	Input Capture.....	1190
27.4.2.2	Output Compare.....	1191
27.4.2.3	Interrupts.....	1192
27.4.2.4	Low Power Mode Behavior.....	1193
27.4.2.5	Debug Mode Behavior.....	1193
27.5	Initialization/ Application Information .....	1193
27.5.1	Selecting the Clock Source .....	1193
27.6	GPT Memory Map/Register Definition.....	1194
27.6.1	GPT Control Register (GPT_CR).....	1195
27.6.2	GPT Prescaler Register (GPT_PR).....	1199
27.6.3	GPT Status Register (GPT_SR).....	1200
27.6.4	GPT Interrupt Register (GPT_IR).....	1201
27.6.5	GPT Output Compare Register 1 (GPT_OCR1).....	1202
27.6.6	GPT Output Compare Register 2 (GPT_OCR2).....	1203
27.6.7	GPT Output Compare Register 3 (GPT_OCR3).....	1203
27.6.8	GPT Input Capture Register 1 (GPT_ICR1).....	1204
27.6.9	GPT Input Capture Register 2 (GPT_ICR2).....	1204
27.6.10	GPT Counter Register (GPT_CNT).....	1205

## Chapter 28

### 2D Graphics Processing Unit (GPU2D)

28.1	Overview.....	1207
28.2	GPU2D Block Diagram.....	1207
28.2.1	R2D GPU.....	1207
28.2.2	V2D GPU.....	1208
28.3	GPU2D Features.....	1209
28.3.1	Full Featured R2D GPU Pipeline.....	1209
28.3.2	Full Featured V2D GPU Pipeline.....	1210

Section number	Title	Page
28.4	GPU2D OPERATIONS.....	1210
28.4.1	R2D GPU Operations.....	1210
28.4.1.1	Line.....	1210
28.4.1.2	Rectangle Fill and Clear.....	1211
28.4.1.3	BitBLT.....	1211
28.4.1.4	Stretch BLT.....	1212
28.4.1.5	Monochrome Expansion and Mask BLT.....	1212
28.4.1.5.1	Monochrome expansion .....	1213
28.4.1.5.2	Mask BLT.....	1213
28.4.1.6	Filter BLT.....	1213
28.4.1.7	R2D Performance of different operations.....	1214
28.4.1.8	Rotation.....	1215
28.4.1.9	Transparency Mode.....	1215
28.4.1.10	Clipping.....	1215
28.4.1.11	R2D GPU Data Formats.....	1215
28.4.1.12	ARGB Data Conversion of R2D GPU.....	1216
28.4.1.13	YUV to RGB Conversion of R2D GPU.....	1216
28.4.1.14	Color Index Input Conversion Support of R2D GPU.....	1217
28.4.1.15	Source/Destination Pre-multiply and De-Multiply Support.....	1217
28.4.1.16	Alpha Blending.....	1218
28.4.1.17	GPU Cache Management.....	1219
28.4.2	V2D GPU Operations.....	1220
28.4.2.1	OPENVG 1.1-API STANDARD for VECTOR GRAPHICS ACCELERATION.....	1220
28.4.2.2	Advantages of Using OpenVG.....	1220
28.4.2.3	OpenVG Target Applications.....	1220
28.4.2.4	OpenVG Features.....	1220
28.4.2.4.1	Core API .....	1220
28.4.2.4.2	The VGU Utility Library .....	1221
28.4.2.4.3	OpenVG Rendering Pipeline.....	1221

Section number	Title	Page
<b>Chapter 29</b>		
<b>I2C Controller (I2C)</b>		
29.1	Overview.....	1223
29.1.1	Features.....	1225
29.1.2	Modes and operations.....	1226
29.2	External Signals.....	1226
29.3	Clocks.....	1227
29.4	Functional description.....	1227
29.4.1	I2C system configuration.....	1227
29.4.2	Arbitration procedure.....	1228
29.4.3	Clock synchronization.....	1228
29.4.4	Handshaking.....	1229
29.4.5	Clock stretching.....	1229
29.4.6	Peripheral bus accesses.....	1230
29.4.7	Generation of transfer error on IP bus.....	1230
29.4.8	Reset.....	1230
29.4.9	Interrupts.....	1230
29.4.10	Byte order.....	1230
29.5	Initialization.....	1231
29.5.1	Initialization sequence.....	1231
29.5.2	Generation of Start.....	1231
29.5.3	Post-transfer software response.....	1231
29.5.4	Generation of Stop.....	1232
29.5.5	Generation of Repeated Start.....	1232
29.5.6	Slave mode.....	1233
29.5.7	Arbitration lost.....	1233
29.6	Software restriction.....	1239
29.7	I2C Memory Map/Register Definition.....	1240
29.7.1	I2C Address Register (I2Cx_IADR).....	1241

Section number	Title	Page
29.7.2	I2C Frequency Divider Register (I2Cx_IFDR).....	1241
29.7.3	I2C Control Register (I2Cx_I2CR).....	1243
29.7.4	I2C Status Register (I2Cx_I2SR).....	1244
29.7.5	I2C Data I/O Register (I2Cx_I2DR).....	1246

## Chapter 30 IOMUX Controller (IOMUXC)

30.1	Overview.....	1247
30.1.1	Features.....	1248
30.2	Clocks.....	1249
30.3	Functional description.....	1249
30.3.1	ALT6 and ALT7 extended muxing modes.....	1250
30.3.2	SW Loopback through SION bit.....	1251
30.3.3	Daisy chain - multi pads driving same module input pin.....	1251
30.4	IOMUXC Memory Map/Register Definition.....	1252
30.4.1	GPR0 (IOMUXC_GPR0).....	1277
30.4.2	GPR1 (IOMUXC_GPR1).....	1279
30.4.3	GPR2 (IOMUXC_GPR2).....	1281
30.4.4	GPR3 (IOMUXC_GPR3).....	1284
30.4.5	GPR4 (IOMUXC_GPR4).....	1288
30.4.6	GPR5 (IOMUXC_GPR5).....	1290
30.4.7	GPR6 (IOMUXC_GPR6).....	1291
30.4.8	GPR7 (IOMUXC_GPR7).....	1291
30.4.9	GPR8 (IOMUXC_GPR8).....	1292
30.4.10	GPR9 (IOMUXC_GPR9).....	1292
30.4.11	GPR10 (IOMUXC_GPR10).....	1293
30.4.12	GPR11 (IOMUXC_GPR11).....	1295
30.4.13	GPR12 (IOMUXC_GPR12).....	1296
30.4.14	GPR13 (IOMUXC_GPR13).....	1297
30.4.15	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_AUD_MCLK).....	1299



Section number	Title	Page
30.4.16	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_AUD_RXC).....	1300
30.4.17	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_AUD_RXD).....	1301
30.4.18	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_AUD_RXFS).....	1302
30.4.19	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_AUD_TXC).....	1303
30.4.20	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_AUD_TXD).....	1304
30.4.21	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_AUD_TXFS).....	1305
30.4.22	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSP11_MISO).....	1306
30.4.23	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSP11_MOSI).....	1307
30.4.24	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSP11_SCLK).....	1308
30.4.25	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSP11_SS0).....	1309
30.4.26	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSP12_MISO).....	1310
30.4.27	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSP12_MOSI).....	1311
30.4.28	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSP12_SCLK).....	1312
30.4.29	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSP12_SS0).....	1313
30.4.30	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR0).....	1314
30.4.31	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR1).....	1315
30.4.32	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA00).....	1316
30.4.33	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA01).....	1317
30.4.34	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA10).....	1318
30.4.35	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA11).....	1319
30.4.36	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA12).....	1320
30.4.37	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA13).....	1321
30.4.38	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA14).....	1322
30.4.39	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA15).....	1323
30.4.40	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA02).....	1324
30.4.41	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA03).....	1325
30.4.42	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA04).....	1326
30.4.43	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA05).....	1327
30.4.44	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA06).....	1328

Section number	Title	Page
30.4.45	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA07).....	1329
30.4.46	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA08).....	1330
30.4.47	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA09).....	1331
30.4.48	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_GDCLK).....	1332
30.4.49	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_GDOE).....	1333
30.4.50	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_GDRL).....	1334
30.4.51	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_GDSP).....	1335
30.4.52	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_COM).....	1336
30.4.53	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL0).....	1337
30.4.54	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL1).....	1338
30.4.55	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL2).....	1339
30.4.56	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL3).....	1340
30.4.57	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_IRQ).....	1341
30.4.58	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_STAT).....	1342
30.4.59	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_WAKE).....	1343
30.4.60	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE0).....	1344
30.4.61	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE1).....	1345
30.4.62	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE2).....	1346
30.4.63	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE3).....	1347
30.4.64	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCLK).....	1348
30.4.65	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_SDLE).....	1349
30.4.66	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_SDOE).....	1350
30.4.67	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_SDSHR).....	1351
30.4.68	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM0).....	1352
30.4.69	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM1).....	1353
30.4.70	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_CRS_DV).....	1354
30.4.71	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_MDC).....	1355
30.4.72	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_MDIO).....	1356
30.4.73	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_REF_CLK).....	1357

Section number	Title	Page
30.4.74	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_RX_ER).....	1358
30.4.75	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA0).....	1359
30.4.76	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA1).....	1360
30.4.77	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_TX_CLK).....	1361
30.4.78	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_TX_EN).....	1362
30.4.79	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA0).....	1363
30.4.80	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA1).....	1364
30.4.81	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_USB_H_DATA).....	1365
30.4.82	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_USB_H_STROBE).....	1366
30.4.83	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C1_SCL).....	1367
30.4.84	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C1_SDA).....	1368
30.4.85	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C2_SCL).....	1369
30.4.86	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C2_SDA).....	1370
30.4.87	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_COL0).....	1371
30.4.88	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_COL1).....	1372
30.4.89	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_COL2).....	1373
30.4.90	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_COL3).....	1374
30.4.91	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_COL4).....	1375
30.4.92	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_COL5).....	1376
30.4.93	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_COL6).....	1377
30.4.94	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_COL7).....	1378
30.4.95	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW0).....	1379
30.4.96	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW1).....	1380
30.4.97	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW2).....	1381
30.4.98	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW3).....	1382
30.4.99	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW4).....	1383
30.4.100	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW5).....	1384
30.4.101	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW6).....	1385
30.4.102	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW7).....	1386

Section number	Title	Page
30.4.103	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_CLK).....	1387
30.4.104	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA00).....	1388
30.4.105	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA01).....	1389
30.4.106	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA10).....	1390
30.4.107	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA11).....	1391
30.4.108	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA12).....	1392
30.4.109	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA13).....	1393
30.4.110	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA14).....	1394
30.4.111	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA15).....	1395
30.4.112	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA16).....	1396
30.4.113	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA17).....	1397
30.4.114	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA18).....	1398
30.4.115	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA19).....	1399
30.4.116	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA02).....	1400
30.4.117	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA20).....	1401
30.4.118	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA21).....	1402
30.4.119	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA22).....	1403
30.4.120	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA23).....	1404
30.4.121	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA03).....	1405
30.4.122	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA04).....	1406
30.4.123	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA05).....	1407
30.4.124	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA06).....	1408
30.4.125	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA07).....	1409
30.4.126	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA08).....	1410
30.4.127	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA09).....	1411
30.4.128	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_ENABLE).....	1412
30.4.129	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_HSYNC).....	1413
30.4.130	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_RESET).....	1414
30.4.131	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_VSYNC).....	1415

Section number	Title	Page
30.4.132	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_PWM1).....	1416
30.4.133	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_REF_CLK_24M).....	1417
30.4.134	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_REF_CLK_32K).....	1418
30.4.135	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_CLK).....	1419
30.4.136	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_CMD).....	1420
30.4.137	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA0).....	1421
30.4.138	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA1).....	1422
30.4.139	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA2).....	1423
30.4.140	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA3).....	1424
30.4.141	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA4).....	1425
30.4.142	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA5).....	1426
30.4.143	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA6).....	1427
30.4.144	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA7).....	1428
30.4.145	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_CLK).....	1429
30.4.146	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_CMD).....	1430
30.4.147	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA0).....	1431
30.4.148	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA1).....	1432
30.4.149	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA2).....	1433
30.4.150	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA3).....	1434
30.4.151	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA4).....	1435
30.4.152	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA5).....	1436
30.4.153	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA6).....	1437
30.4.154	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA7).....	1438
30.4.155	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_RESET).....	1439
30.4.156	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD3_CLK).....	1440
30.4.157	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD3_CMD).....	1441
30.4.158	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD3_DATA0).....	1442
30.4.159	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD3_DATA1).....	1443
30.4.160	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD3_DATA2).....	1444

Section number	Title	Page
30.4.161	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD3_DATA3).....	1445
30.4.162	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART1_RXD).....	1446
30.4.163	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART1_TXD).....	1447
30.4.164	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_WDOG_B).....	1448
30.4.165	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_AUD_MCLK).....	1449
30.4.166	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_AUD_RXC).....	1451
30.4.167	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_AUD_RXD).....	1453
30.4.168	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_AUD_RXFS).....	1455
30.4.169	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_AUD_TXC).....	1457
30.4.170	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_AUD_TXD).....	1459
30.4.171	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_AUD_TXFS).....	1461
30.4.172	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR00).....	1463
30.4.173	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR01).....	1465
30.4.174	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR10).....	1467
30.4.175	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR11).....	1469
30.4.176	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR12).....	1471
30.4.177	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR13).....	1473
30.4.178	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR14).....	1475
30.4.179	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR15).....	1477
30.4.180	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR02).....	1479
30.4.181	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR03).....	1481
30.4.182	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR04).....	1483
30.4.183	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR05).....	1485
30.4.184	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR06).....	1487
30.4.185	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR07).....	1489
30.4.186	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR08).....	1491
30.4.187	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR09).....	1493
30.4.188	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_CAS_B).....	1495
30.4.189	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_CS0_B).....	1497

Section number	Title	Page
30.4.190	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_CS1_B).....	1499
30.4.191	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM0).....	1501
30.4.192	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM1).....	1503
30.4.193	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM2).....	1505
30.4.194	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM3).....	1507
30.4.195	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_RAS_B).....	1509
30.4.196	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_RESET).....	1511
30.4.197	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDBA0).....	1513
30.4.198	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDBA1).....	1515
30.4.199	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDBA2).....	1517
30.4.200	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCKE0).....	1519
30.4.201	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCKE1).....	1521
30.4.202	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCLK0_P).....	1523
30.4.203	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ODT0).....	1525
30.4.204	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ODT1).....	1527
30.4.205	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS0_P).....	1529
30.4.206	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS1_P).....	1531
30.4.207	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS2_P).....	1533
30.4.208	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS3_P).....	1535
30.4.209	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDWE_B).....	1537
30.4.210	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSP11_MISO).....	1539
30.4.211	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSP11_MOSI).....	1541
30.4.212	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSP11_SCLK).....	1543
30.4.213	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSP11_SS0).....	1545
30.4.214	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSP12_MISO).....	1547
30.4.215	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSP12_MOSI).....	1549
30.4.216	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSP12_SCLK).....	1551
30.4.217	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSP12_SS0).....	1553
30.4.218	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_BDR0).....	1555

Section number	Title	Page
30.4.219	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_BDR1).....	1557
30.4.220	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA00).....	1559
30.4.221	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA01).....	1561
30.4.222	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA10).....	1563
30.4.223	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA11).....	1565
30.4.224	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA12).....	1567
30.4.225	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA13).....	1569
30.4.226	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA14).....	1571
30.4.227	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA15).....	1573
30.4.228	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA02).....	1575
30.4.229	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA03).....	1577
30.4.230	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA04).....	1579
30.4.231	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA05).....	1581
30.4.232	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA06).....	1583
30.4.233	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA07).....	1585
30.4.234	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA08).....	1587
30.4.235	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA09).....	1589
30.4.236	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_GDCLK).....	1591
30.4.237	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_GDOE).....	1593
30.4.238	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_GDRL).....	1595
30.4.239	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_GDSP).....	1597
30.4.240	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_PWR_COM).....	1599
30.4.241	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_PWR_CTRL0).....	1601
30.4.242	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_PWR_CTRL1).....	1603
30.4.243	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_PWR_CTRL2).....	1605
30.4.244	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_PWR_CTRL3).....	1607
30.4.245	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_PWR_IRQ).....	1609
30.4.246	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_PWR_STAT).....	1611
30.4.247	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_PWR_WAKE).....	1613



Section number	Title	Page
30.4.248	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_SDCE0).....	1615
30.4.249	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_SDCE1).....	1617
30.4.250	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_SDCE2).....	1619
30.4.251	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_SDCE3).....	1621
30.4.252	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_SDCLK).....	1623
30.4.253	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_SDLE).....	1625
30.4.254	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_SDOE).....	1627
30.4.255	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_SDSHR).....	1629
30.4.256	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_VCOM0).....	1631
30.4.257	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_VCOM1).....	1633
30.4.258	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_CRSDV).....	1635
30.4.259	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_MDC).....	1637
30.4.260	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_MDIO).....	1639
30.4.261	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_REF_CLK).....	1641
30.4.262	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_RX_ER).....	1643
30.4.263	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_RX_DATA0).....	1645
30.4.264	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_RX_DATA1).....	1647
30.4.265	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_TX_CLK).....	1649
30.4.266	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_TX_EN).....	1651
30.4.267	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_TX_DATA0).....	1653
30.4.268	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_TX_DATA1).....	1655
30.4.269	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_USB_H_DATA).....	1657
30.4.270	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_USB_H_STROBE).....	1659
30.4.271	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C1_SCL).....	1661
30.4.272	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C1_SDA).....	1663
30.4.273	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C2_SCL).....	1665
30.4.274	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C2_SDA).....	1667
30.4.275	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_MOD).....	1669
30.4.276	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TCK).....	1671

Section number	Title	Page
30.4.277	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TDI).....	1673
30.4.278	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TDO).....	1675
30.4.279	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TMS).....	1677
30.4.280	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TRSTB).....	1679
30.4.281	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_COL0).....	1681
30.4.282	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_COL1).....	1683
30.4.283	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_COL2).....	1685
30.4.284	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_COL3).....	1687
30.4.285	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_COL4).....	1689
30.4.286	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_COL5).....	1691
30.4.287	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_COL6).....	1693
30.4.288	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_COL7).....	1695
30.4.289	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW0).....	1697
30.4.290	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW1).....	1699
30.4.291	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW2).....	1701
30.4.292	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW3).....	1703
30.4.293	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW4).....	1705
30.4.294	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW5).....	1707
30.4.295	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW6).....	1709
30.4.296	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW7).....	1711
30.4.297	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_CLK).....	1713
30.4.298	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA00).....	1715
30.4.299	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA01).....	1717
30.4.300	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA10).....	1719
30.4.301	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA11).....	1721
30.4.302	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA12).....	1723
30.4.303	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA13).....	1725
30.4.304	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA14).....	1727
30.4.305	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA15).....	1729

Section number	Title	Page
30.4.306	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA16).....	1731
30.4.307	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA17).....	1733
30.4.308	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA18).....	1735
30.4.309	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA19).....	1737
30.4.310	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA02).....	1739
30.4.311	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA20).....	1741
30.4.312	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA21).....	1743
30.4.313	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA22).....	1745
30.4.314	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA23).....	1747
30.4.315	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA03).....	1749
30.4.316	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA04).....	1751
30.4.317	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA05).....	1753
30.4.318	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA06).....	1755
30.4.319	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA07).....	1757
30.4.320	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA08).....	1759
30.4.321	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA09).....	1761
30.4.322	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_ENABLE).....	1763
30.4.323	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_HSYNC).....	1765
30.4.324	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_RESET).....	1767
30.4.325	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_VSYNC).....	1769
30.4.326	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_PWM1).....	1771
30.4.327	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_REF_CLK_24M).....	1773
30.4.328	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_REF_CLK_32K).....	1775
30.4.329	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_CLK).....	1777
30.4.330	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_CMD).....	1779
30.4.331	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA0).....	1781
30.4.332	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA1).....	1783
30.4.333	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA2).....	1785
30.4.334	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA3).....	1787

Section number	Title	Page
30.4.335	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA4).....	1789
30.4.336	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA5).....	1791
30.4.337	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA6).....	1793
30.4.338	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA7).....	1795
30.4.339	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_CLK).....	1797
30.4.340	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_CMD).....	1799
30.4.341	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA0).....	1801
30.4.342	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA1).....	1803
30.4.343	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA2).....	1805
30.4.344	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA3).....	1807
30.4.345	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA4).....	1809
30.4.346	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA5).....	1811
30.4.347	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA6).....	1813
30.4.348	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA7).....	1815
30.4.349	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_RESET).....	1817
30.4.350	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD3_CLK).....	1819
30.4.351	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD3_CMD).....	1821
30.4.352	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD3_DATA0).....	1823
30.4.353	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD3_DATA1).....	1825
30.4.354	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD3_DATA2).....	1827
30.4.355	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD3_DATA3).....	1829
30.4.356	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART1_RXD).....	1831
30.4.357	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART1_TXD).....	1833
30.4.358	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_WDOG_B).....	1835
30.4.359	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_ADDDS).....	1836
30.4.360	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_DDRMODE_CTL).....	1837
30.4.361	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_DDRPKE).....	1838
30.4.362	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_DDRPK).....	1839
30.4.363	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_DDRHYS).....	1840

Section number	Title	Page
30.4.364	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_DDRMODE).....	1841
30.4.365	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_B0DS).....	1842
30.4.366	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_CTLDS).....	1842
30.4.367	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_B1DS).....	1843
30.4.368	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE).....	1844
30.4.369	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_B2DS).....	1845
30.4.370	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_B3DS).....	1845
30.4.371	Select Input Register (IOMUXC_ANALOG_USB_OTG_ID_SELECT_INPUT).....	1846
30.4.372	Select Input Register (IOMUXC_ANALOG_USB_H1_ID_SELECT_INPUT).....	1847
30.4.373	Select Input Register (IOMUXC_AUD4_INPUT_DA_AMX_SELECT_INPUT).....	1847
30.4.374	Select Input Register (IOMUXC_AUD4_INPUT_DB_AMX_SELECT_INPUT).....	1848
30.4.375	Select Input Register (IOMUXC_AUD4_INPUT_RXCLK_AMX_SELECT_INPUT).....	1849
30.4.376	Select Input Register (IOMUXC_AUD4_INPUT_RXFS_AMX_SELECT_INPUT).....	1849
30.4.377	Select Input Register (IOMUXC_AUD4_INPUT_TXCLK_AMX_SELECT_INPUT).....	1850
30.4.378	Select Input Register (IOMUXC_AUD4_INPUT_TXFS_AMX_SELECT_INPUT).....	1851
30.4.379	Select Input Register (IOMUXC_AUD5_INPUT_DA_AMX_SELECT_INPUT).....	1851
30.4.380	Select Input Register (IOMUXC_AUD5_INPUT_DB_AMX_SELECT_INPUT).....	1852
30.4.381	Select Input Register (IOMUXC_AUD5_INPUT_RXCLK_AMX_SELECT_INPUT).....	1853
30.4.382	Select Input Register (IOMUXC_AUD5_INPUT_RXFS_AMX_SELECT_INPUT).....	1854
30.4.383	Select Input Register (IOMUXC_AUD5_INPUT_TXCLK_AMX_SELECT_INPUT).....	1855
30.4.384	Select Input Register (IOMUXC_AUD5_INPUT_TXFS_AMX_SELECT_INPUT).....	1856
30.4.385	Select Input Register (IOMUXC_AUD6_INPUT_DA_AMX_SELECT_INPUT).....	1857
30.4.386	Select Input Register (IOMUXC_AUD6_INPUT_DB_AMX_SELECT_INPUT).....	1858
30.4.387	Select Input Register (IOMUXC_AUD6_INPUT_RXCLK_AMX_SELECT_INPUT).....	1859
30.4.388	Select Input Register (IOMUXC_AUD6_INPUT_RXFS_AMX_SELECT_INPUT).....	1860
30.4.389	Select Input Register (IOMUXC_AUD6_INPUT_TXCLK_AMX_SELECT_INPUT).....	1861
30.4.390	Select Input Register (IOMUXC_AUD6_INPUT_TXFS_AMX_SELECT_INPUT).....	1862
30.4.391	Select Input Register (IOMUXC_CCM_PMIC_READY_SELECT_INPUT).....	1862
30.4.392	Select Input Register (IOMUXC_CSI_CSI_DATA00_SELECT_INPUT).....	1863

Section number	Title	Page
30.4.393	Select Input Register (IOMUXC_CSI_CSI_DATA01_SELECT_INPUT).....	1864
30.4.394	Select Input Register (IOMUXC_CSI_CSI_DATA02_SELECT_INPUT).....	1864
30.4.395	Select Input Register (IOMUXC_CSI_CSI_DATA03_SELECT_INPUT).....	1865
30.4.396	Select Input Register (IOMUXC_CSI_CSI_DATA04_SELECT_INPUT).....	1866
30.4.397	Select Input Register (IOMUXC_CSI_CSI_DATA05_SELECT_INPUT).....	1866
30.4.398	Select Input Register (IOMUXC_CSI_CSI_DATA06_SELECT_INPUT).....	1867
30.4.399	Select Input Register (IOMUXC_CSI_CSI_DATA07_SELECT_INPUT).....	1868
30.4.400	Select Input Register (IOMUXC_CSI_CSI_DATA08_SELECT_INPUT).....	1868
30.4.401	Select Input Register (IOMUXC_CSI_CSI_DATA09_SELECT_INPUT).....	1869
30.4.402	Select Input Register (IOMUXC_CSI_CSI_DATA10_SELECT_INPUT).....	1870
30.4.403	Select Input Register (IOMUXC_CSI_CSI_DATA11_SELECT_INPUT).....	1870
30.4.404	Select Input Register (IOMUXC_CSI_CSI_DATA12_SELECT_INPUT).....	1871
30.4.405	Select Input Register (IOMUXC_CSI_CSI_DATA13_SELECT_INPUT).....	1872
30.4.406	Select Input Register (IOMUXC_CSI_CSI_DATA14_SELECT_INPUT).....	1873
30.4.407	Select Input Register (IOMUXC_CSI_CSI_DATA15_SELECT_INPUT).....	1874
30.4.408	Select Input Register (IOMUXC_CSI_CSI_HSYNC_SELECT_INPUT).....	1874
30.4.409	Select Input Register (IOMUXC_CSI_CSI_PIXCLK_SELECT_INPUT).....	1875
30.4.410	Select Input Register (IOMUXC_CSI_CSI_VSYNC_SELECT_INPUT).....	1876
30.4.411	Select Input Register (IOMUXC_ECSP11_CSPI_CLK_IN_SELECT_INPUT).....	1876
30.4.412	Select Input Register (IOMUXC_ECSP11_DATAREADY_B_SELECT_INPUT).....	1877
30.4.413	Select Input Register (IOMUXC_ECSP11_MISO_SELECT_INPUT).....	1878
30.4.414	Select Input Register (IOMUXC_ECSP11_MOSI_SELECT_INPUT).....	1879
30.4.415	Select Input Register (IOMUXC_ECSP11_SS0_SELECT_INPUT).....	1880
30.4.416	Select Input Register (IOMUXC_ECSP11_SS1_SELECT_INPUT).....	1881
30.4.417	Select Input Register (IOMUXC_ECSP11_SS2_SELECT_INPUT).....	1882
30.4.418	Select Input Register (IOMUXC_ECSP11_SS3_SELECT_INPUT).....	1883
30.4.419	Select Input Register (IOMUXC_ECSP12_CSPI_CLK_IN_SELECT_INPUT).....	1883
30.4.420	Select Input Register (IOMUXC_ECSP12_MISO_SELECT_INPUT).....	1884
30.4.421	Select Input Register (IOMUXC_ECSP12_MOSI_SELECT_INPUT).....	1885

Section number	Title	Page
30.4.422	Select Input Register (IOMUXC_ECSPi2_SS0_SELECT_INPUT).....	1885
30.4.423	Select Input Register (IOMUXC_ECSPi2_SS1_SELECT_INPUT).....	1886
30.4.424	Select Input Register (IOMUXC_ECSPi3_CSPI_CLK_IN_SELECT_INPUT).....	1887
30.4.425	Select Input Register (IOMUXC_ECSPi3_DATAREADY_B_SELECT_INPUT).....	1887
30.4.426	Select Input Register (IOMUXC_ECSPi3_MISO_SELECT_INPUT).....	1888
30.4.427	Select Input Register (IOMUXC_ECSPi3_MOSI_SELECT_INPUT).....	1889
30.4.428	Select Input Register (IOMUXC_ECSPi3_SS0_SELECT_INPUT).....	1889
30.4.429	Select Input Register (IOMUXC_ECSPi3_SS1_SELECT_INPUT).....	1890
30.4.430	Select Input Register (IOMUXC_ECSPi3_SS2_SELECT_INPUT).....	1891
30.4.431	Select Input Register (IOMUXC_ECSPi3_SS3_SELECT_INPUT).....	1892
30.4.432	Select Input Register (IOMUXC_ECSPi4_CSPI_CLK_IN_SELECT_INPUT).....	1892
30.4.433	Select Input Register (IOMUXC_ECSPi4_MISO_SELECT_INPUT).....	1893
30.4.434	Select Input Register (IOMUXC_ECSPi4_MOSI_SELECT_INPUT).....	1894
30.4.435	Select Input Register (IOMUXC_ECSPi4_SS0_SELECT_INPUT).....	1894
30.4.436	Select Input Register (IOMUXC_ECSPi4_SS1_SELECT_INPUT).....	1895
30.4.437	Select Input Register (IOMUXC_ECSPi4_SS2_SELECT_INPUT).....	1896
30.4.438	Select Input Register (IOMUXC_EPDC_EPDC_PWR_IRQ_SELECT_INPUT).....	1897
30.4.439	Select Input Register (IOMUXC_EPDC_EPDC_PWR_STAT_SELECT_INPUT).....	1898
30.4.440	Select Input Register (IOMUXC_FEC_FEC_COL_SELECT_INPUT).....	1898
30.4.441	Select Input Register (IOMUXC_FEC_FEC_MDI_SELECT_INPUT).....	1899
30.4.442	Select Input Register (IOMUXC_FEC_FEC_RX_DATA0_SELECT_INPUT).....	1900
30.4.443	Select Input Register (IOMUXC_FEC_FEC_RX_DATA1_SELECT_INPUT).....	1900
30.4.444	Select Input Register (IOMUXC_FEC_FEC_RX_CLK_SELECT_INPUT).....	1901
30.4.445	Select Input Register (IOMUXC_FEC_FEC_RX_DV_SELECT_INPUT).....	1902
30.4.446	Select Input Register (IOMUXC_FEC_FEC_RX_ER_SELECT_INPUT).....	1902
30.4.447	Select Input Register (IOMUXC_FEC_FEC_TX_CLK_SELECT_INPUT).....	1903
30.4.448	Select Input Register (IOMUXC_GPT_CAPIN1_SELECT_INPUT).....	1904
30.4.449	Select Input Register (IOMUXC_GPT_CAPIN2_SELECT_INPUT).....	1905
30.4.450	Select Input Register (IOMUXC_GPT_CLKIN_SELECT_INPUT).....	1906

Section number	Title	Page
30.4.451	Select Input Register (IOMUXC_I2C1_SCL_IN_SELECT_INPUT).....	1906
30.4.452	Select Input Register (IOMUXC_I2C1_SDA_IN_SELECT_INPUT).....	1907
30.4.453	Select Input Register (IOMUXC_I2C2_SCL_IN_SELECT_INPUT).....	1908
30.4.454	Select Input Register (IOMUXC_I2C2_SDA_IN_SELECT_INPUT).....	1908
30.4.455	Select Input Register (IOMUXC_I2C3_SCL_IN_SELECT_INPUT).....	1909
30.4.456	Select Input Register (IOMUXC_I2C3_SDA_IN_SELECT_INPUT).....	1910
30.4.457	Select Input Register (IOMUXC_KEY_COL0_SELECT_INPUT).....	1910
30.4.458	Select Input Register (IOMUXC_KEY_COL1_SELECT_INPUT).....	1911
30.4.459	Select Input Register (IOMUXC_KEY_COL2_SELECT_INPUT).....	1912
30.4.460	Select Input Register (IOMUXC_KEY_COL3_SELECT_INPUT).....	1912
30.4.461	Select Input Register (IOMUXC_KEY_COL4_SELECT_INPUT).....	1913
30.4.462	Select Input Register (IOMUXC_KEY_COL5_SELECT_INPUT).....	1914
30.4.463	Select Input Register (IOMUXC_KEY_COL6_SELECT_INPUT).....	1914
30.4.464	Select Input Register (IOMUXC_KEY_COL7_SELECT_INPUT).....	1915
30.4.465	Select Input Register (IOMUXC_KEY_ROW0_SELECT_INPUT).....	1916
30.4.466	Select Input Register (IOMUXC_KEY_ROW1_SELECT_INPUT).....	1916
30.4.467	Select Input Register (IOMUXC_KEY_ROW2_SELECT_INPUT).....	1917
30.4.468	Select Input Register (IOMUXC_KEY_ROW3_SELECT_INPUT).....	1918
30.4.469	Select Input Register (IOMUXC_KEY_ROW4_SELECT_INPUT).....	1918
30.4.470	Select Input Register (IOMUXC_KEY_ROW5_SELECT_INPUT).....	1919
30.4.471	Select Input Register (IOMUXC_KEY_ROW6_SELECT_INPUT).....	1920
30.4.472	Select Input Register (IOMUXC_KEY_ROW7_SELECT_INPUT).....	1920
30.4.473	Select Input Register (IOMUXC_LCD_BUSY_SELECT_INPUT).....	1921
30.4.474	Select Input Register (IOMUXC_LCD_DATA00_SELECT_INPUT).....	1922
30.4.475	Select Input Register (IOMUXC_LCD_DATA01_SELECT_INPUT).....	1923
30.4.476	Select Input Register (IOMUXC_LCD_DATA02_SELECT_INPUT).....	1924
30.4.477	Select Input Register (IOMUXC_LCD_DATA03_SELECT_INPUT).....	1925
30.4.478	Select Input Register (IOMUXC_LCD_DATA04_SELECT_INPUT).....	1926
30.4.479	Select Input Register (IOMUXC_LCD_DATA05_SELECT_INPUT).....	1927



Section number	Title	Page
30.4.480	Select Input Register (IOMUXC_LCD_DATA06_SELECT_INPUT).....	1928
30.4.481	Select Input Register (IOMUXC_LCD_DATA07_SELECT_INPUT).....	1929
30.4.482	Select Input Register (IOMUXC_LCD_DATA08_SELECT_INPUT).....	1930
30.4.483	Select Input Register (IOMUXC_LCD_DATA09_SELECT_INPUT).....	1931
30.4.484	Select Input Register (IOMUXC_LCD_DATA10_SELECT_INPUT).....	1932
30.4.485	Select Input Register (IOMUXC_LCD_DATA11_SELECT_INPUT).....	1933
30.4.486	Select Input Register (IOMUXC_LCD_DATA12_SELECT_INPUT).....	1934
30.4.487	Select Input Register (IOMUXC_LCD_DATA13_SELECT_INPUT).....	1935
30.4.488	Select Input Register (IOMUXC_LCD_DATA14_SELECT_INPUT).....	1936
30.4.489	Select Input Register (IOMUXC_LCD_DATA15_SELECT_INPUT).....	1937
30.4.490	Select Input Register (IOMUXC_LCD_DATA16_SELECT_INPUT).....	1938
30.4.491	Select Input Register (IOMUXC_LCD_DATA17_SELECT_INPUT).....	1939
30.4.492	Select Input Register (IOMUXC_LCD_DATA18_SELECT_INPUT).....	1940
30.4.493	Select Input Register (IOMUXC_LCD_DATA19_SELECT_INPUT).....	1941
30.4.494	Select Input Register (IOMUXC_LCD_DATA20_SELECT_INPUT).....	1942
30.4.495	Select Input Register (IOMUXC_LCD_DATA21_SELECT_INPUT).....	1943
30.4.496	Select Input Register (IOMUXC_LCD_DATA22_SELECT_INPUT).....	1944
30.4.497	Select Input Register (IOMUXC_LCD_DATA23_SELECT_INPUT).....	1945
30.4.498	Select Input Register (IOMUXC_SPDIF_SPDIF_IN1_SELECT_INPUT).....	1945
30.4.499	Select Input Register (IOMUXC_SPDIF_TX_CLK2_SELECT_INPUT).....	1946
30.4.500	Select Input Register (IOMUXC_UART1_UART_RTS_B_SELECT_INPUT).....	1947
30.4.501	Select Input Register (IOMUXC_UART1_UART_RX_DATA_SELECT_INPUT).....	1948
30.4.502	Select Input Register (IOMUXC_UART2_UART_RTS_B_SELECT_INPUT).....	1948
30.4.503	Select Input Register (IOMUXC_UART2_UART_RX_DATA_SELECT_INPUT).....	1949
30.4.504	Select Input Register (IOMUXC_UART3_UART_RTS_B_SELECT_INPUT).....	1950
30.4.505	Select Input Register (IOMUXC_UART3_UART_RX_DATA_SELECT_INPUT).....	1950
30.4.506	Select Input Register (IOMUXC_UART4_UART_RTS_B_SELECT_INPUT).....	1951
30.4.507	Select Input Register (IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT).....	1952
30.4.508	Select Input Register (IOMUXC_UART5_UART_RTS_B_SELECT_INPUT).....	1952

Section number	Title	Page
30.4.509	Select Input Register (IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT).....	1953
30.4.510	Select Input Register (IOMUXC_USB_OTG2_OC_SELECT_INPUT).....	1954
30.4.511	Select Input Register (IOMUXC_USB_OTG1_OC_SELECT_INPUT).....	1954
30.4.512	Select Input Register (IOMUXC_USDHC1_CARD_DET_SELECT_INPUT).....	1955
30.4.513	Select Input Register (IOMUXC_USDHC1_WP_ON_SELECT_INPUT).....	1956
30.4.514	Select Input Register (IOMUXC_USDHC2_CARD_DET_SELECT_INPUT).....	1956
30.4.515	Select Input Register (IOMUXC_USDHC2_WP_ON_SELECT_INPUT).....	1957
30.4.516	Select Input Register (IOMUXC_USDHC3_CARD_DET_SELECT_INPUT).....	1958
30.4.517	Select Input Register (IOMUXC_USDHC3_DATA4_IN_SELECT_INPUT).....	1958
30.4.518	Select Input Register (IOMUXC_USDHC3_DATA5_IN_SELECT_INPUT).....	1959
30.4.519	Select Input Register (IOMUXC_USDHC3_DATA6_IN_SELECT_INPUT).....	1960
30.4.520	Select Input Register (IOMUXC_USDHC3_DATA7_IN_SELECT_INPUT).....	1961
30.4.521	Select Input Register (IOMUXC_USDHC3_WP_ON_SELECT_INPUT).....	1961
30.4.522	Select Input Register (IOMUXC_USDHC4_CARD_CLK_IN_SELECT_INPUT).....	1962
30.4.523	Select Input Register (IOMUXC_USDHC4_CARD_DET_SELECT_INPUT).....	1963
30.4.524	Select Input Register (IOMUXC_USDHC4_CMD_IN_SELECT_INPUT).....	1963
30.4.525	Select Input Register (IOMUXC_USDHC4_DATA0_IN_SELECT_INPUT).....	1964
30.4.526	Select Input Register (IOMUXC_USDHC4_DATA1_IN_SELECT_INPUT).....	1965
30.4.527	Select Input Register (IOMUXC_USDHC4_DATA2_IN_SELECT_INPUT).....	1965
30.4.528	Select Input Register (IOMUXC_USDHC4_DATA3_IN_SELECT_INPUT).....	1966
30.4.529	Select Input Register (IOMUXC_USDHC4_DATA4_IN_SELECT_INPUT).....	1967
30.4.530	Select Input Register (IOMUXC_USDHC4_DATA5_IN_SELECT_INPUT).....	1967
30.4.531	Select Input Register (IOMUXC_USDHC4_DATA6_IN_SELECT_INPUT).....	1968
30.4.532	Select Input Register (IOMUXC_USDHC4_DATA7_IN_SELECT_INPUT).....	1969
30.4.533	Select Input Register (IOMUXC_USDHC4_WP_ON_SELECT_INPUT).....	1969
30.4.534	Select Input Register (IOMUXC_EIM_DTACK_B_SELECT_INPUT).....	1970
30.4.535	Select Input Register (IOMUXC_EIM_WAIT_B_SELECT_INPUT).....	1971

Section number	Title	Page
<b>Chapter 31</b>		
<b>Keypad Port (KPP)</b>		
31.1	Overview .....	1973
31.1.1	Features.....	1975
31.1.2	Modes and Operations.....	1975
31.2	Clocks.....	1975
31.3	External Signals.....	1975
31.3.1	Input Pins.....	1977
31.3.2	Output Pins.....	1977
31.3.3	Generation of Transfer Error Signal on Peripheral Bus.....	1978
31.4	Functional Description.....	1978
31.4.1	Keypad Matrix Construction.....	1978
31.4.2	Keypad Port Configuration.....	1979
31.4.3	Keypad Matrix Scanning.....	1979
31.4.4	Keypad Standby.....	1979
31.4.5	Glitch Suppression on Keypad Inputs.....	1980
31.4.6	Multiple Key Closures.....	1982
31.4.6.1	Ghost Key Problem and Correction.....	1984
31.4.7	3-Point Contact Keys Support.....	1986
31.5	Initialization/Application Information.....	1987
31.5.1	Typical Keypad Configuration and Scanning Sequence.....	1988
31.5.2	Key Press Interrupt Scanning Sequence.....	1988
31.5.3	Additional Comments.....	1988
31.6	KPP Memory Map/Register Definition.....	1989
31.6.1	Keypad Control Register (KPP_KPCR).....	1989
31.6.2	Keypad Status Register (KPP_KPSR).....	1990
31.6.3	Keypad Data Direction Register (KPP_KDDR).....	1992
31.6.4	Keypad Data Register (KPP_KPDR).....	1992

## Chapter 32

### Multi Mode DDR Controller (MMDC)

32.1	Overview.....	1995
32.1.1	MMDC feature summary.....	1996
32.2	External Signals.....	1999
32.3	Clocks.....	2000
32.4	Functional Description.....	2000
32.4.1	Write/Read data flow.....	2000
32.4.1.1	Write data flow.....	2000
32.4.1.2	Read data flow.....	2001
32.4.2	MMDC initialization .....	2001
32.4.3	Configuring the MMDC registers.....	2003
32.4.4	MMDC Address Space.....	2003
32.4.4.1	Address decoding .....	2003
32.4.4.2	Chip select settings.....	2006
32.4.4.2.1	Creating 4 Gbyte address space with 2 Gbyte CS density.....	2006
32.4.4.2.2	Creating 2 Gbyte address spaces with 1 Gbyte CS density.....	2007
32.4.4.3	Translation of AXI accesses to DDR accessess.....	2007
32.4.4.3.1	Example .....	2008
32.4.4.4	Address mirroring .....	2008
32.4.5	LPDDR2 and DDR3 pin mux mapping.....	2009
32.4.6	Power Saving and Clock Frequency Change modes.....	2010
32.4.6.1	Power saving general.....	2010
32.4.6.2	Self refresh and Frequency change entry/exit.....	2011
32.4.7	Reset .....	2013
32.4.7.1	Hard reset.....	2013
32.4.7.2	Warm reset.....	2013
32.4.7.3	Software reset .....	2014
32.4.8	Refresh Scheme.....	2014

Section number	Title	Page
32.4.9	Burst Length options towards DDR .....	2015
32.4.10	Exclusive accesses handling.....	2015
32.4.11	AXI Error Handling.....	2017
32.5	Performance.....	2017
32.5.1	Arbitration and reordering mechanism.....	2017
32.5.1.1	Arbitration General.....	2017
32.5.1.2	Real time channel mode.....	2018
32.5.1.3	Dynamic scoring mode (Arbitration Winning Conditions).....	2018
32.5.1.4	Guarding (aging) mechanism.....	2019
32.5.2	Prediction mechanism.....	2019
32.5.3	Special Optimization for accesses towards DDR3.....	2020
32.6	MMDC Debug .....	2021
32.6.1	Hardware debug monitor.....	2021
32.6.2	Step By Step (SBS) software monitor.....	2021
32.7	MMDC Profiling.....	2022
32.8	LPDDR2 Refresh Rate Update and Timing Derating.....	2023
32.9	DLL Off mode.....	2024
32.10	ODT Configuration .....	2025
32.11	Calibration Process.....	2026
32.11.1	delay-line.....	2027
32.11.2	ZQ calibration .....	2028
32.11.2.1	ZQ automatic (hardware) calibration process.....	2029
32.11.2.1.1	ZQ automatic Pull-up calibration.....	2029
32.11.2.1.2	ZQ automatic Pull-down calibration.....	2030
32.11.2.2	ZQ software calibration process.....	2030
32.11.2.3	ZQ calibration commands .....	2030
32.11.3	Read DQS Gating Calibration.....	2031
32.11.3.1	Hardware DQS Gating Calibration.....	2031
32.11.3.1.1	Hardware DQS Calibration with MPR.....	2031

Section number	Title	Page
32.11.3.1.2	Hardware DQS Calibration with pre-defined value.....	2032
32.11.3.2	SW read DQS gating Calibration.....	2034
32.11.3.2.1	SW read Calibration with MPR.....	2034
32.11.3.2.2	SW read Calibration with pre-defined value.....	2035
32.11.4	Read Calibration.....	2037
32.11.4.1	Hardware (automatic) Read Calibration.....	2037
32.11.4.1.1	Hardware (automatic) Calibration with MPR (DDR3) /DQ Calibration (LPDDR2).....	2038
32.11.4.1.2	Hardware (automatic) Calibration with pre-defined value.....	2038
32.11.4.2	SW Read Calibration.....	2040
32.11.4.2.1	Calibration with MPR(DDR3)/DQ calibration(LPDDR2).....	2040
32.11.4.2.2	Calibration with pre-defined value.....	2040
32.11.5	Write Calibration.....	2042
32.11.5.1	HW (automatic) Write Calibration.....	2042
32.11.5.2	SW Write Calibration.....	2043
32.11.6	Write leveling Calibration.....	2045
32.11.6.1	Hardware Write Leveling Calibration.....	2046
32.11.6.2	SW Write Leveling Calibration.....	2047
32.11.7	Write fine tuning.....	2048
32.11.8	Read fine tuning.....	2049
32.12	MMDC Memory Map/Register Definition.....	2049
32.12.1	MMDC Core Control Register (MMDC_MDCTL).....	2053
32.12.2	MMDC Core Power Down Control Register (MMDC_MDPDC).....	2055
32.12.3	MMDC Core ODT Timing Control Register (MMDC_MDOTC).....	2057
32.12.4	MMDC Core Timing Configuration Register 0 (MMDC_MDCFG0).....	2059
32.12.5	MMDC Core Timing Configuration Register 1 (MMDC_MDCFG1).....	2061
32.12.6	MMDC Core Timing Configuration Register 2 (MMDC_MDCFG2).....	2063
32.12.7	MMDC Core Miscellaneous Register (MMDC_MDMISC).....	2065
32.12.8	MMDC Core Special Command Register (MMDC_MDSCR).....	2068

Section number	Title	Page
32.12.9	MMDC Core Refresh Control Register (MMDC_MDREF).....	2071
32.12.10	MMDC Core Read/Write Command Delay Register (MMDC_MDRWD).....	2073
32.12.11	MMDC Core Out of Reset Delays Register (MMDC_MDOR).....	2075
32.12.12	MMDC Core MRR Data Register (MMDC_MDMRR).....	2076
32.12.13	MMDC Core Timing Configuration Register 3 (MMDC_MDCFG3LP).....	2077
32.12.14	MMDC Core MR4 Derating Register (MMDC_MDMR4).....	2079
32.12.15	MMDC Core Address Space Partition Register (MMDC_MDASP).....	2081
32.12.16	MMDC Core AXI Reordering Control Regsiter (MMDC_MAARCR).....	2082
32.12.17	MMDC Core Power Saving Control and Status Register (MMDC_MAPSR).....	2084
32.12.18	MMDC Core Exclusive ID Monitor Register0 (MMDC_MAEXIDR0).....	2086
32.12.19	MMDC Core Exclusive ID Monitor Register1 (MMDC_MAEXIDR1).....	2087
32.12.20	MMDC Core Debug and Profiling Control Register 0 (MMDC_MADPCR0).....	2088
32.12.21	MMDC Core Debug and Profiling Control Register 1 (MMDC_MADPCR1).....	2089
32.12.22	MMDC Core Debug and Profiling Status Register 0 (MMDC_MADPSR0).....	2090
32.12.23	MMDC Core Debug and Profiling Status Register 1 (MMDC_MADPSR1).....	2090
32.12.24	MMDC Core Debug and Profiling Status Register 2 (MMDC_MADPSR2).....	2091
32.12.25	MMDC Core Debug and Profiling Status Register 3 (MMDC_MADPSR3).....	2091
32.12.26	MMDC Core Debug and Profiling Status Register 4 (MMDC_MADPSR4).....	2092
32.12.27	MMDC Core Debug and Profiling Status Register 5 (MMDC_MADPSR5).....	2092
32.12.28	MMDC Core Step By Step Address Register (MMDC_MASBS0).....	2093
32.12.29	MMDC Core Step By Step Address Attributes Register (MMDC_MASBS1).....	2093
32.12.30	MMDC Core General Purpose Register (MMDC_MAGENP).....	2094
32.12.31	MMDC PHY ZQ HW control register (MMDC_MPZQHWCTRL).....	2095
32.12.32	MMDC PHY ZQ SW control register (MMDC_MPZQSWCTRL).....	2098
32.12.33	MMDC PHY Write Leveling Configuration and Error Status Register (MMDC_MPWLGCRC).....	2100
32.12.34	MMDC PHY Write Leveling Delay Control Register 0 (MMDC_MPWLDECTRL0).....	2103
32.12.35	MMDC PHY Write Leveling Delay Control Register 1 (MMDC_MPWLDECTRL1).....	2105
32.12.36	MMDC PHY Write Leveling delay-line Status Register (MMDC_MPWLDLST).....	2108
32.12.37	MMDC PHY ODT control register (MMDC_MPODTCTRL).....	2110

Section number	Title	Page
32.12.38	MMDC PHY Read DQ Byte0 Delay Register (MMDC_MPRDDQBY0DL).....	2112
32.12.39	MMDC PHY Read DQ Byte1 Delay Register (MMDC_MPRDDQBY1DL).....	2115
32.12.40	MMDC PHY Read DQ Byte2 Delay Register (MMDC_MPRDDQBY2DL).....	2118
32.12.41	MMDC PHY Read DQ Byte3 Delay Register (MMDC_MPRDDQBY3DL).....	2120
32.12.42	MMDC PHY Write DQ Byte0 Delay Register (MMDC_MPWRDQBY0DL).....	2123
32.12.43	MMDC PHY Write DQ Byte1 Delay Register (MMDC_MPWRDQBY1DL).....	2125
32.12.44	MMDC PHY Write DQ Byte2 Delay Register (MMDC_MPWRDQBY2DL).....	2127
32.12.45	MMDC PHY Write DQ Byte3 Delay Register (MMDC_MPWRDQBY3DL).....	2130
32.12.46	MMDC PHY Read DQS Gating Control Register 0 (MMDC_MPDGCTRL0).....	2132
32.12.47	MMDC PHY Read DQS Gating Control Register 1 (MMDC_MPDGCTRL1).....	2134
32.12.48	MMDC PHY Read DQS Gating delay-line Status Register (MMDC_MPDGDLST0).....	2137
32.12.49	MMDC PHY Read delay-lines Configuration Register (MMDC_MPRDDLCTL).....	2138
32.12.50	MMDC PHY Read delay-lines Status Register (MMDC_MPRDDLST).....	2140
32.12.51	MMDC PHY Write delay-lines Configuration Register (MMDC_MPWRDLCTL).....	2141
32.12.52	MMDC PHY Write delay-lines Status Register (MMDC_MPWRDLST).....	2142
32.12.53	MMDC PHY CK Control Register (MMDC_MPSDCTRL).....	2143
32.12.54	MMDC ZQ LPDDR2 HW Control Register (MMDC_MPZQLP2CTL).....	2144
32.12.55	MMDC PHY Read Delay HW Calibration Control Register (MMDC_MPRDDLHWCTL).....	2146
32.12.56	MMDC PHY Write Delay HW Calibration Control Register (MMDC_MPWRDLHWCTL).....	2148
32.12.57	MMDC PHY Read Delay HW Calibration Status Register 0 (MMDC_MPRDDLHWST0).....	2149
32.12.58	MMDC PHY Read Delay HW Calibration Status Register 1 (MMDC_MPRDDLHWST1).....	2150
32.12.59	MMDC PHY Write Delay HW Calibration Status Register 0 (MMDC_MPWRDLHWST0).....	2151
32.12.60	MMDC PHY Write Delay HW Calibration Status Register 1 (MMDC_MPWRDLHWST1).....	2152
32.12.61	MMDC PHY Write Leveling HW Error Register (MMDC_MPWLHWERR).....	2153
32.12.62	MMDC PHY Read DQS Gating HW Status Register 0 (MMDC_MPDGHWST0).....	2153
32.12.63	MMDC PHY Read DQS Gating HW Status Register 1 (MMDC_MPDGHWST1).....	2154
32.12.64	MMDC PHY Read DQS Gating HW Status Register 2 (MMDC_MPDGHWST2).....	2154
32.12.65	MMDC PHY Read DQS Gating HW Status Register 3 (MMDC_MPDGHWST3).....	2155
32.12.66	MMDC PHY Pre-defined Compare Register 1 (MMDC_MPPDCMPR1).....	2156



Section number	Title	Page
32.12.67	MMDC PHY Pre-defined Compare and CA delay-line Configuration Register (MMDC_MPPDCMPR2).....	2157
32.12.68	MMDC PHY SW Dummy Access Register (MMDC_MPSWDAR0).....	2159
32.12.69	MMDC PHY SW Dummy Read Data Register 0 (MMDC_MPSWDRDR0).....	2160
32.12.70	MMDC PHY SW Dummy Read Data Register 1 (MMDC_MPSWDRDR1).....	2161
32.12.71	MMDC PHY SW Dummy Read Data Register 2 (MMDC_MPSWDRDR2).....	2161
32.12.72	MMDC PHY SW Dummy Read Data Register 3 (MMDC_MPSWDRDR3).....	2161
32.12.73	MMDC PHY SW Dummy Read Data Register 4 (MMDC_MPSWDRDR4).....	2162
32.12.74	MMDC PHY SW Dummy Read Data Register 5 (MMDC_MPSWDRDR5).....	2162
32.12.75	MMDC PHY SW Dummy Read Data Register 6 (MMDC_MPSWDRDR6).....	2163
32.12.76	MMDC PHY SW Dummy Read Data Register 7 (MMDC_MPSWDRDR7).....	2163
32.12.77	MMDC PHY Measure Unit Register (MMDC_MPMUR0).....	2164
32.12.78	MMDC Write CA delay-line controller (MMDC_MPWRCADL).....	2165
32.12.79	MMDC Duty Cycle Control Register (MMDC_MPDCCR).....	2167

## Chapter 33

### Network Interconnect Bus System (NIC-301)

33.1	Overview .....	2169
33.2	Block diagram.....	2169
33.2.1	NIC-301 Main Features.....	2170
33.2.2	Modes and Operations.....	2170
33.3	External Signals.....	2170
33.4	Memory Map and Register Definition.....	2170
33.4.1	Memory Map.....	2171
33.4.2	Configuration programmers model.....	2171
33.4.2.1	Address control and ID registers.....	2172
33.4.2.2	AMBA master interface block (AMIB) configuration registers.....	2172
33.4.2.3	ASIB (AMBA slave interface block) configuration registers.....	2172
33.4.3	Register Descriptions.....	2173
33.4.4	NIC Specific Parameters.....	2173

## Chapter 34

### On-Chip OTP Controller (OCOTP\_CTRL)

34.1	Overview.....	2175
34.1.1	Features.....	2175
34.2	Clocks.....	2175
34.3	Top-Level Symbol and Functional Overview.....	2176
34.3.1	Operation.....	2176
34.3.1.1	Shadow Register Reload.....	2177
34.3.1.2	Fuse and Shadow register read.....	2177
34.3.1.3	Fuse and Shadow Register Writes.....	2178
34.3.1.4	Write Postamble.....	2179
34.3.2	Fuse Shadow Memory Footprint.....	2180
34.3.3	OTP Read/Write Timing Parameters.....	2182
34.3.4	Hardware Visible Fuses.....	2183
34.3.5	Behavior During Reset.....	2183
34.3.6	Secure JTAG control.....	2183
34.4	Fuse Map.....	2184
34.5	OCOTP Memory Map/Register Definition.....	2184
34.5.1	OTP Controller Control Register (OCOTP_CTRLn).....	2188
34.5.2	OTP Controller Timing Register (OCOTP_TIMING).....	2190
34.5.3	OTP Controller Write Data Register (OCOTP_DATA).....	2190
34.5.4	OTP Controller Write Data Register (OCOTP_READ_CTRL).....	2191
34.5.5	OTP Controller Read Data Register (OCOTP_READ_FUSE_DATA).....	2192
34.5.6	Sticky bit Register (OCOTP_SW_STICKY).....	2192
34.5.7	Software Controllable Signals Register (OCOTP_SCSn).....	2194
34.5.8	OTP Controller CRC test address (OCOTP_CRC_ADDR).....	2195
34.5.9	OTP Controller CRC Value Register (OCOTP_CRC_VALUE).....	2195
34.5.10	OTP Controller Timing Register (OCOTP_UMC_TIMING).....	2196
34.5.11	OTP Controller Version Register (OCOTP_VERSION).....	2196

Section number	Title	Page
34.5.12	Value of OTP Bank0 Word0 (Lock controls) (OCOTP_LOCK).....	2197
34.5.13	Value of OTP Bank0 Word1 (Configuration and Manufacturing Info.) (OCOTP_CFG0).....	2200
34.5.14	Value of OTP Bank0 Word2 (Configuration and Manufacturing Info.) (OCOTP_CFG1).....	2200
34.5.15	Value of OTP Bank0 Word3 (Configuration and Manufacturing Info.) (OCOTP_CFG2).....	2201
34.5.16	Value of OTP Bank0 Word4 (Configuration and Manufacturing Info.) (OCOTP_CFG3).....	2201
34.5.17	Value of OTP Bank0 Word5 (Configuration and Manufacturing Info.) (OCOTP_CFG4).....	2202
34.5.18	Value of OTP Bank0 Word6 (Configuration and Manufacturing Info.) (OCOTP_CFG5).....	2202
34.5.19	Value of OTP Bank0 Word7 (Configuration and Manufacturing Info.) (OCOTP_CFG6).....	2203
34.5.20	Value of OTP Bank1 Word0 (Memory Related Info.) (OCOTP_MEM0).....	2203
34.5.21	Value of OTP Bank1 Word1 (Memory Related Info.) (OCOTP_MEM1).....	2204
34.5.22	Value of OTP Bank1 Word2 (Memory Related Info.) (OCOTP_MEM2).....	2204
34.5.23	Value of OTP Bank1 Word3 (Memory Related Info.) (OCOTP_MEM3).....	2205
34.5.24	Value of OTP Bank1 Word4 (Memory Related Info.) (OCOTP_MEM4).....	2205
34.5.25	Value of OTP Bank1 Word5 (Memory Related Info.) (OCOTP_ANA0).....	2206
34.5.26	Value of OTP Bank1 Word6 (General Purpose Customer Defined Info.) (OCOTP_ANA1).....	2206
34.5.27	Value of OTP Bank1 Word7 (General Purpose Customer Defined Info.) (OCOTP_ANA2).....	2207
34.5.28	Shadow Register for OTP Bank2 Word0 (DCP and CRYPTO Key) (OCOTP_DCP0).....	2207
34.5.29	Shadow Register for OTP Bank2 Word1 (DCP and CRYPTO Key) (OCOTP_DCP1).....	2208
34.5.30	Shadow Register for OTP Bank2 Word2 (DCP and CRYPTO Key) (OCOTP_DCP2).....	2208
34.5.31	Shadow Register for OTP Bank2 Word3 (DCP and CRYPTO Key) (OCOTP_DCP3).....	2209
34.5.32	Shadow Register for OTP Bank2 Word4 (DCP Key) (OCOTP_DCP4).....	2209
34.5.33	Shadow Register for OTP Bank2 Word5 (DCP Key) (OCOTP_DCP5).....	2210
34.5.34	Shadow Register for OTP Bank2 Word6 (DCP Key) (OCOTP_DCP6).....	2210
34.5.35	Shadow Register for OTP Bank2 Word7 (DCP Key) (OCOTP_DCP7).....	2211
34.5.36	Shadow Register for OTP Bank3 Word0 (SRK Hash) (OCOTP_SRK0).....	2211
34.5.37	Shadow Register for OTP Bank3 Word1 (SRK Hash) (OCOTP_SRK1).....	2212
34.5.38	Shadow Register for OTP Bank3 Word2 (SRK Hash) (OCOTP_SRK2).....	2212
34.5.39	Shadow Register for OTP Bank3 Word3 (SRK Hash) (OCOTP_SRK3).....	2213
34.5.40	Shadow Register for OTP Bank3 Word4 (SRK Hash) (OCOTP_SRK4).....	2213

Section number	Title	Page
34.5.41	Shadow Register for OTP Bank3 Word5 (SRK Hash) (OCOTP_SRK5).....	2214
34.5.42	Shadow Register for OTP Bank3 Word6 (SRK Hash) (OCOTP_SRK6).....	2214
34.5.43	Shadow Register for OTP Bank3 Word7 (SRK Hash) (OCOTP_SRK7).....	2215
34.5.44	Value of OTP Bank4 Word0 (Secure JTAG Response Field) (OCOTP_SJC_RESP0).....	2215
34.5.45	Value of OTP Bank4 Word1 (Secure JTAG Response Field) (OCOTP_SJC_RESP1).....	2216
34.5.46	Value of OTP Bank4 Word2 (MAC Address) (OCOTP_MAC0).....	2217
34.5.47	Value of OTP Bank4 Word3 (MAC Address) (OCOTP_MAC1).....	2217
34.5.48	Value of OTP Bank4 Word4 (HW Capabilities) (OCOTP_CRC0).....	2218
34.5.49	Value of OTP Bank4 Word5 (HW Capabilities) (OCOTP_CRC1).....	2218
34.5.50	Value of OTP Bank4 Word6 (HW Capabilities) (OCOTP_GP1).....	2219
34.5.51	Value of OTP Bank4 Word7 (HW Capabilities) (OCOTP_GP2).....	2219
34.5.52	Value of OTP Bank5 Word0 (HW Capabilities) (OCOTP_SW_GP0).....	2220
34.5.53	Value of OTP Bank5 Word1 (HW Capabilities) (OCOTP_SW_GP1).....	2220
34.5.54	Value of OTP Bank5 Word2 (HW Capabilities) (OCOTP_SW_GP2).....	2221
34.5.55	Value of OTP Bank5 Word3 (HW Capabilities) (OCOTP_SW_GP3).....	2221
34.5.56	Value of OTP Bank5 Word4 (HW Capabilities) (OCOTP_SW_GP4).....	2222
34.5.57	Value of OTP Bank5 Word5 (HW Capabilities) (OCOTP_MISC_CONF).....	2222
34.5.58	Value of OTP Bank5 Word6 (HW Capabilities) (OCOTP_FIELD_RETURN).....	2223
34.5.59	Value of OTP Bank5 Word7 (HW Capabilities) (OCOTP_SRK_REVOKE).....	2223
34.5.60	Value of OTP Bank6 Word0 (HW Capabilities) (OCOTP_GP_LO0).....	2224
34.5.61	Value of OTP Bank6 Word1 (HW Capabilities) (OCOTP_GP_LO1).....	2224
34.5.62	Value of OTP Bank6 Word2 (HW Capabilities) (OCOTP_GP_LO2).....	2225
34.5.63	Value of OTP Bank6 Word3 (HW Capabilities) (OCOTP_GP_LO3).....	2225
34.5.64	Value of OTP Bank6 Word4 (HW Capabilities) (OCOTP_GP_LO4).....	2226
34.5.65	Value of OTP Bank6 Word5 (HW Capabilities) (OCOTP_GP_LO5).....	2226
34.5.66	Value of OTP Bank6 Word6 (HW Capabilities) (OCOTP_GP_LO6).....	2227
34.5.67	Value of OTP Bank6 Word7 (HW Capabilities) (OCOTP_GP_LO7).....	2227
34.5.68	Value of OTP Bank7 Word0 (HW Capabilities) (OCOTP_GP_HI0).....	2228
34.5.69	Value of OTP Bank7 Word1 (HW Capabilities) (OCOTP_GP_HI1).....	2228

Section number	Title	Page
34.5.70	Value of OTP Bank7 Word2 (HW Capabilities) (OCOTP_GP_HI2).....	2229
34.5.71	Value of OTP Bank7 Word3 (HW Capabilities) (OCOTP_GP_HI3).....	2229
34.5.72	Value of OTP Bank7 Word4 (HW Capabilities) (OCOTP_GP_HI4).....	2230
34.5.73	Value of OTP Bank7 Word5 (HW Capabilities) (OCOTP_GP_HI5).....	2230
34.5.74	Value of OTP Bank7 Word6 (HW Capabilities) (OCOTP_GP_HI6).....	2231
34.5.75	Value of OTP Bank7 Word7 (HW Capabilities) (OCOTP_GP_HI7).....	2231

## Chapter 35 On-Chip RAM Memory Controller (OCRAM)

35.1	Overview.....	2233
35.2	Basic Functions.....	2234
35.2.1	Read/Write Arbitration.....	2234
35.2.2	TrustZone.....	2235
35.3	Advanced Features.....	2235
35.3.1	Read Data Wait State.....	2235
35.3.2	Read Address Pipeline.....	2236
35.3.3	Write Data Pipeline.....	2236
35.3.4	Write Address Pipeline.....	2237
35.4	Programmable Registers.....	2237

## Chapter 36 Power Management Unit (PMU)

36.1	Overview.....	2239
36.2	Digital LDO Regulators.....	2241
36.3	Analog LDO Regulators.....	2242
36.3.1	LDO 1P1.....	2242
36.3.2	LDO 2P5.....	2243
36.3.2.1	Low Power Operation.....	2243
36.4	USB LDO Regulator.....	2244
36.5	SNVS Regulator.....	2244

Section number	Title	Page
36.6	Power Modes.....	2244
36.6.1	Reverse Well Biasing.....	2245
36.7	PMU Memory Map/Register Definition.....	2245
36.7.1	Digital Regulator Core Register (PMU_REG_CORE).....	2246
36.7.2	Miscellaneous Register 0 (PMU_MISC0).....	2249
36.7.3	Miscellaneous Register 1 (PMU_MISC1n).....	2252
36.7.4	Miscellaneous Control Register (PMU_MISC2n).....	2255

## Chapter 37

### Pulse Width Modulation (PWM)

37.1	Overview.....	2259
37.2	External Signals.....	2260
37.3	Clocks.....	2261
37.4	Functional Description.....	2262
37.4.1	Operation.....	2263
37.4.1.1	FIFO.....	2263
37.4.1.2	Rollover and Compare Event.....	2264
37.4.1.3	Low Power Mode Behavior.....	2264
37.4.1.4	Debug Mode Behavior.....	2264
37.5	Enable Sequence for the PWM.....	2264
37.6	Disable Sequence for the PWM.....	2265
37.7	PWM Memory Map/Register Definition.....	2265
37.7.1	PWM Control Register (PWMx_PWMCR).....	2267
37.7.2	PWM Status Register (PWMx_PWMSR).....	2269
37.7.3	PWM Interrupt Register (PWMx_PWMIR).....	2270
37.7.4	PWM Sample Register (PWMx_PWMSAR).....	2271
37.7.5	PWM Period Register (PWMx_PWMPR).....	2272
37.7.6	PWM Counter Register (PWMx_PWMCNR).....	2273

Section number	Title	Page
<b>Chapter 38</b>		
<b>Pixel Pipeline (PXP)</b>		
38.1	Overview.....	2275
38.2	Clocks.....	2276
38.3	Top-level architecture.....	2276
38.3.1	Processing Details.....	2278
38.3.2	Scaling Operation.....	2279
38.3.3	Decimation Image Scaling.....	2280
38.3.4	Bilinear Image Scaling Filter .....	2282
38.3.5	YUV 4:2:2 Image Scaling.....	2284
38.3.6	YUV 4:2:0 Image Scaling.....	2285
38.3.7	RGB/YUV444 Image Scaling.....	2287
38.3.8	Color Space Conversion (CSC).....	2287
38.3.9	CSC1 Operation.....	2288
38.3.10	YUV versus YCbCr Support.....	2289
38.3.11	CSC2 operation.....	2289
38.3.12	Alpha Blending/Color Key .....	2289
38.3.13	Alpha Blend.....	2290
38.3.14	Color Key.....	2291
38.3.15	LUT.....	2291
38.3.16	Lookup Modes.....	2292
38.3.17	DIRECT_Y8.....	2292
38.3.18	DIRECT_RGB444.....	2292
38.3.19	DIRECT_RGB454.....	2292
38.3.20	CACHE_RGB565.....	2293
38.3.21	Output Modes.....	2294
38.3.22	Y8 .....	2294
38.3.23	RGBW4444CFA.....	2295
38.3.23.1	CFA Correction.....	2295

Section number	Title	Page
38.3.24	RGB888.....	2296
38.3.25	Rotation.....	2296
38.3.26	Output Buffer.....	2299
38.3.27	Address calculator.....	2299
38.3.28	Block size selection.....	2299
38.3.29	Interlaced Video Support.....	2299
38.3.30	LCDIF Handshake.....	2300
38.3.31	LCDIF Abort.....	2303
38.3.32	Theory of Operation.....	2303
38.3.33	Pixel Handling.....	2304
38.3.34	Output Buffer Composition.....	2305
38.3.35	PS Image Processing.....	2305
38.3.36	Letterboxing.....	2306
38.3.37	Clipping source images.....	2306
38.3.38	Color Key Processing.....	2308
38.3.39	In Place Processing (PS buffer is destination buffer).....	2310
38.3.40	Alpha Surface (AS) Processing.....	2310
38.3.41	Alpha Handling.....	2310
38.3.42	Color Key Processing (AS_CTRL).....	2310
38.4	Output Image Processing.....	2311
38.4.1	Output Image Size.....	2311
38.4.2	Output Format.....	2311
38.4.3	Rotation/Flip operations.....	2311
38.5	Queuing PXP transactions.....	2312
38.6	Error Handling.....	2312
38.6.1	Known PXP Limitations/Issues.....	2313
38.7	PXP Memory Map/Register Definition.....	2313
38.7.1	Control Register 0 (PXP_CTRL).....	2316
38.7.2	Status Register (PXP_STAT).....	2318



Section number	Title	Page
38.7.3	Output Buffer Control Register (PXP_OUT_CTRL).....	2320
38.7.4	Output Frame Buffer Pointer (PXP_OUT_BUF).....	2322
38.7.5	Output Frame Buffer Pointer #2 (PXP_OUT_BUF2).....	2323
38.7.6	Output Buffer Pitch (PXP_OUT_PITCH).....	2323
38.7.7	Output Surface Lower Right Coordinate (PXP_OUT_LRC).....	2324
38.7.8	Processed Surface Upper Left Coordinate (PXP_OUT_PS_ULC).....	2325
38.7.9	Processed Surface Lower Right Coordinate (PXP_OUT_PS_LRC).....	2326
38.7.10	Alpha Surface Upper Left Coordinate (PXP_OUT_AS_ULC).....	2327
38.7.11	Alpha Surface Lower Right Coordinate (PXP_OUT_AS_LRC).....	2328
38.7.12	Processed Surface (PS) Control Register (PXP_PS_CTRL).....	2329
38.7.13	PS Input Buffer Address (PXP_PS_BUF).....	2331
38.7.14	PS U/Cb or 2 Plane UV Input Buffer Address (PXP_PS_UBUF).....	2331
38.7.15	PS V/Cr Input Buffer Address (PXP_PS_VBUF).....	2332
38.7.16	Processed Surface Pitch (PXP_PS_PITCH).....	2333
38.7.17	PS Background Color (PXP_PS_BACKGROUND).....	2334
38.7.18	PS Scale Factor Register (PXP_PS_SCALE).....	2334
38.7.19	PS Scale Offset Register (PXP_PS_OFFSET).....	2336
38.7.20	PS Color Key Low (PXP_PS_CLRKEYLOW).....	2336
38.7.21	PS Color Key High (PXP_PS_CLRKEYHIGH).....	2337
38.7.22	Alpha Surface Control (PXP_AS_CTRL).....	2338
38.7.23	Alpha Surface Buffer Pointer (PXP_AS_BUF).....	2340
38.7.24	Alpha Surface Pitch (PXP_AS_PITCH).....	2341
38.7.25	Overlay Color Key Low (PXP_AS_CLRKEYLOW).....	2341
38.7.26	Overlay Color Key High (PXP_AS_CLRKEYHIGH).....	2342
38.7.27	Color Space Conversion Coefficient Register 0 (PXP_CSC1_COEF0).....	2343
38.7.28	Color Space Conversion Coefficient Register 1 (PXP_CSC1_COEF1).....	2344
38.7.29	Color Space Conversion Coefficient Register 2 (PXP_CSC1_COEF2).....	2345
38.7.30	Color Space Conversion Control Register. (PXP_CSC2_CTRL).....	2346
38.7.31	Color Space Conversion Coefficient Register 0 (PXP_CSC2_COEF0).....	2347

Section number	Title	Page
38.7.32	Color Space Conversion Coefficient Register 1 (PXP_CSC2_COEF1).....	2348
38.7.33	Color Space Conversion Coefficient Register 2 (PXP_CSC2_COEF2).....	2348
38.7.34	Color Space Conversion Coefficient Register 3 (PXP_CSC2_COEF3).....	2349
38.7.35	Color Space Conversion Coefficient Register 4 (PXP_CSC2_COEF4).....	2350
38.7.36	Color Space Conversion Coefficient Register 5 (PXP_CSC2_COEF5).....	2350
38.7.37	Lookup Table Control Register. (PXP_LUT_CTRL).....	2351
38.7.38	Lookup Table Control Register. (PXP_LUT_ADDR).....	2353
38.7.39	Lookup Table Data Register. (PXP_LUT_DATA).....	2355
38.7.40	Lookup Table External Memory Address Register. (PXP_LUT_EXTMEM).....	2355
38.7.41	Color Filter Array Register. (PXP_CFA).....	2356
38.7.42	Histogram Control Register. (PXP_HIST_CTRL).....	2356
38.7.43	2-level Histogram Parameter Register. (PXP_HIST2_PARAM).....	2357
38.7.44	4-level Histogram Parameter Register. (PXP_HIST4_PARAM).....	2358
38.7.45	8-level Histogram Parameter 0 Register. (PXP_HIST8_PARAM0).....	2359
38.7.46	8-level Histogram Parameter 1 Register. (PXP_HIST8_PARAM1).....	2360
38.7.47	16-level Histogram Parameter 0 Register. (PXP_HIST16_PARAM0).....	2361
38.7.48	16-level Histogram Parameter 1 Register. (PXP_HIST16_PARAM1).....	2362
38.7.49	16-level Histogram Parameter 2 Register. (PXP_HIST16_PARAM2).....	2363
38.7.50	16-level Histogram Parameter 3 Register. (PXP_HIST16_PARAM3).....	2364
38.7.51	PXP Power Control Register. (PXP_POWER).....	2365
38.7.52	Next Frame Pointer (PXP_NEXT).....	2366

## Chapter 39

### Random Number Generator (RNGB)

39.1	Introduction.....	2369
39.1.1	Block Diagram.....	2369
39.1.2	Features.....	2370
39.2	Modes of Operation.....	2370
39.2.1	Self Test Mode.....	2370
39.2.2	Seed Generation Mode.....	2370

Section number	Title	Page
39.2.3	Random Number Generation Mode.....	2371
39.2.4	Verification Mode.....	2371
39.3	Memory Map/Register Definition.....	2371
39.3.1	RNGB Version ID Register (RNG_VER).....	2372
39.3.2	RNGB Command Register (RNG_CMD).....	2373
39.3.3	RNGB Control Register (RNG_CR).....	2374
39.3.4	RNGB Status Register (RNG_SR).....	2376
39.3.5	RNGB Error Status Register (RNG_ESR).....	2378
39.3.6	RNGB Output FIFO (RNG_OUT).....	2380
39.3.7	RNGB Entropy Register (RNG_ER).....	2380
39.3.8	Verification Control Register (RNG_VCR).....	2381
39.3.9	XKEY Data (RNG_XKEY).....	2382
39.3.10	Oscillator Counter Control Register (RNG_OCCR).....	2383
39.3.11	Oscillator Counter (RNG_OSC_CNT).....	2383
39.3.12	Oscillator Counter Status (RNG_OSC_CNT_STAT).....	2384
39.4	Functional Description.....	2384
39.4.1	Pseudorandom Number Generator (PRNG).....	2385
39.4.2	True Random Number Generator (TRNG).....	2385
39.4.3	Resets.....	2385
39.4.3.1	Power-on/Hardware Reset.....	2386
39.4.3.2	Software Reset.....	2386
39.4.4	RNG Interrupts.....	2386
39.5	Initialization/Application Information.....	2387
39.5.1	Manual Seeding.....	2387
39.5.2	Automatic Seeding.....	2388
39.5.3	Deterministic Mode.....	2388
39.5.4	Oscillator Frequency Verification.....	2388

Section number	Title	Page
<b>Chapter 40</b>		
<b>ROM Controller with Patch (ROMC)</b>		
40.1	Overview.....	2391
40.1.1	Features.....	2392
40.1.2	Modes of Operation.....	2392
40.1.2.1	Low Power Mode.....	2393
40.2	Clocks.....	2393
40.3	Memory Map.....	2393
40.3.1	ROM Memory Map in detail.....	2393
40.4	Functional Description.....	2394
40.4.1	ROM Controller (ROMC) Functional Description.....	2394
40.4.1.1	Functionality overview.....	2394
40.4.2	ROMC Functional Description.....	2394
40.4.2.1	ROMC Disabling.....	2395
40.4.2.2	ROMC Event Priority.....	2395
40.4.2.3	Data Fixing.....	2395
40.4.2.4	Opcode Patching.....	2396
40.4.2.4.1	Typical Software Response to Opcode Patch.....	2397
40.4.2.5	External Boot Feature.....	2398
40.4.2.6	Alternate Masters and ROMC.....	2399
40.5	ROMCP Memory Map/Register Definition.....	2399
40.5.1	ROMC Data Registers (ROMC_ROMPATCHnD).....	2400
40.5.2	ROMC Control Register (ROMC_ROMPATCHCNTL).....	2401
40.5.3	ROMC Enable Register High (ROMC_ROMPATCHENH).....	2402
40.5.4	ROMC Enable Register Low (ROMC_ROMPATCHENL).....	2402
40.5.5	ROMC Address Registers (ROMC_ROMPATCHnA).....	2403
40.5.6	ROMC Status Register (ROMC_ROMPATCHSR).....	2404

## Chapter 41

### Smart Direct Memory Access Controller (SDMA)

41.1	Overview.....	2407
41.1.1	Block Diagram.....	2407
41.1.2	Features.....	2409
41.2	External Signals.....	2411
41.3	Clocks.....	2411
41.4	Functional Description.....	2411
41.4.1	SDMA Core.....	2414
41.4.1.1	SDMA Core Structure.....	2414
41.4.1.2	Program Control Unit (PCU).....	2417
41.4.1.2.1	Instruction Types.....	2417
41.4.1.2.2	PCU States.....	2418
41.4.1.3	SDMA Core Memory.....	2421
41.4.2	Scheduler.....	2421
41.4.2.1	Primary Functions.....	2421
41.4.2.2	Channels and DMA Requests.....	2422
41.4.2.2.1	Channels.....	2422
41.4.2.2.2	DMA Requests.....	2422
41.4.2.2.3	Mapping from DMA Requests to Channels and Priorities.....	2422
41.4.2.3	Scheduler Functional Description.....	2422
41.4.2.3.1	Scheduler Overview.....	2422
41.4.2.3.2	DMA Requests Scanning.....	2423
41.4.2.3.3	Mapping DMA Requests to Pending Channels.....	2424
41.4.2.3.4	Channel Overflow.....	2427
41.4.2.3.5	Runnable Channels Evaluation.....	2427
41.4.2.3.6	Next Channel Decision Tree.....	2429
41.4.2.3.7	Scheduler State Diagram.....	2431
41.4.2.3.8	Scheduler Pipeline Timing Diagram.....	2433

Section number	Title	Page
41.4.2.3.9	Channel-DMA Request Mapping.....	2433
41.4.2.3.10	Examples: How to Start a Channel.....	2433
41.4.2.4	Context Switching.....	2434
41.4.2.4.1	Context Switch Modes.....	2435
41.4.2.4.2	Context Switch Procedure.....	2435
41.4.2.4.3	Context Map in Memory.....	2437
41.4.3	Functional Units.....	2437
41.4.3.1	Burst DMA Unit.....	2437
41.4.3.1.1	Burst DMA Structure.....	2438
41.4.3.1.2	Burst DMA Registers.....	2439
41.4.3.1.3	Burst DMA Data Transfers.....	2440
41.4.3.1.3.1	Data Retrieval from the ARM platform Memory.....	2440
41.4.3.1.3.2	Storing Data Into the ARM platform Memory.....	2440
41.4.3.1.3.3	Transferring Data Between Two ARM platform Memory Locations-Burst DMA Unit.....	2441
41.4.3.2	Peripheral DMA Unit.....	2441
41.4.3.2.1	Peripheral DMA Structure.....	2442
41.4.3.2.2	Peripheral DMA Registers.....	2443
41.4.3.2.3	Peripheral DMA Data Transfers.....	2444
41.4.3.2.3.1	Data Retrieval from the ARM platform Memory or Peripheral.....	2444
41.4.3.2.3.2	Storing Data into the ARM platform Memory or Peripheral...	2444
41.4.3.2.3.3	Transferring Data Between Two ARM platform Memory Locations-Peripheral DMA Unit.....	2445
41.4.4	SDMA Security Support.....	2445
41.4.4.1	Locked Mode.....	2445
41.4.5	OnCE and PCU Debug States.....	2446
41.4.6	SDMA Clocks and Low Power Modes.....	2448
41.4.6.1	Clock Gating and Low Power Modes.....	2449
41.4.6.1.1	Coarse Clock Gating.....	2449

Section number	Title	Page
41.4.6.1.2	Refined Clock Gating.....	2450
41.4.6.1.3	Low Power Modes and User Control.....	2450
41.4.6.1.3.1	SLEEP Mode.....	2451
41.4.6.1.3.2	RUN Mode.....	2451
41.4.6.1.3.3	DEBUG Mode.....	2452
41.4.6.1.4	Stop Mode Response.....	2452
41.4.6.2	Reset.....	2452
41.4.7	Software Interface.....	2452
41.4.8	Initialization Information.....	2453
41.4.8.1	Hardware Reset.....	2453
41.4.8.2	Channel Script Execution.....	2454
41.4.8.3	Initialization and Script Execution Setup Sequence.....	2454
41.4.9	SDMA Programming Model.....	2455
41.4.9.1	State and Registers Per Channel.....	2455
41.4.9.2	General Purpose Registers.....	2456
41.4.9.3	Functional Unit State.....	2456
41.4.9.3.1	Program Counter Register (PC).....	2456
41.4.9.3.2	Flags.....	2456
41.4.9.3.3	Return Program Counter (RPC).....	2457
41.4.9.3.4	Loop Mode Start Program Counter (SPC).....	2457
41.4.9.3.5	Loop Mode End Program Counter (EPC).....	2457
41.4.9.4	Context Switching-Programming.....	2458
41.4.9.5	Address Space.....	2459
41.4.9.5.1	Instruction Memory Map.....	2460
41.4.9.5.2	Data Memory Map.....	2460
41.4.10	SDMA Initialization.....	2462
41.4.10.1	Hardware Reset-SDMA.....	2462
41.4.10.2	Standard Boot Sequence.....	2462
41.4.10.3	User-Defined Boot Sequence.....	2463

Section number	Title	Page
41.4.10.4	Script Loading and Context Initialization.....	2463
41.4.11	Instruction Description.....	2464
41.4.11.1	Scheduling Instructions.....	2464
41.4.11.2	Conditional Branch Instructions.....	2464
41.4.11.3	Unconditional Jump Instructions.....	2465
41.4.11.4	Subroutine Return Instructions.....	2465
41.4.11.5	Loop Instruction.....	2465
41.4.11.6	Miscellaneous Instructions.....	2466
41.4.11.7	Logic Instructions.....	2466
41.4.11.8	Arithmetic Instructions.....	2466
41.4.11.9	Compare Instructions.....	2467
41.4.11.10	Test Instructions.....	2467
41.4.11.11	Byte Permutation Instructions.....	2467
41.4.11.12	Bit Shift Instructions.....	2468
41.4.11.13	Bit Manipulation Instructions.....	2468
41.4.11.14	SDMA Memory Access Instructions.....	2468
41.4.11.15	Functional Unit Instructions.....	2469
41.4.11.16	Illegal Instructions.....	2469
41.4.11.17	Debug Instructions.....	2469
41.4.12	Functional Units Programming Model.....	2470
41.4.12.1	Burst DMA Unit Programming.....	2471
41.4.12.1.1	Memory Source Address Register (MSA).....	2471
41.4.12.1.2	Memory Destination Address Register (MDA).....	2472
41.4.12.1.3	Memory Data Buffer Register (MD).....	2472
41.4.12.1.4	State Register (MS).....	2473
41.4.12.1.5	Burst DMA Write (stf).....	2474
41.4.12.1.6	Burst DMA Read (ldf).....	2477
41.4.12.1.7	Prefetch/Flush and Auto-Flush Management-Burst DMA Unit.....	2478



Section number	Title	Page
41.4.12.1.8	Data Alignment and Endianness-Burst DMA Unit.....	2480
41.4.12.1.8.1	Burst DMA in Read Mode.....	2480
41.4.12.1.8.2	Burst DMA in Write Mode.....	2481
41.4.12.1.8.3	Endianness-Burst DMA Unit.....	2483
41.4.12.1.9	Burst DMA Unit Copy Mode.....	2483
41.4.12.1.10	Burst DMA Unit Error Management.....	2484
41.4.12.1.11	Conditional Yielding-Burst DMA Unit.....	2486
41.4.12.2	Peripheral DMA Unit Programming.....	2487
41.4.12.2.1	Peripheral Source Address Register (PSA).....	2488
41.4.12.2.2	Peripheral Destination Address Register (PDA).....	2488
41.4.12.2.3	Peripheral Data Register (PD).....	2489
41.4.12.2.4	Peripheral State Register (PS).....	2489
41.4.12.2.5	Peripheral DMA Write (stf)-Write Mode.....	2491
41.4.12.2.6	Peripheral DMA Read (ldf)-Read Mode.....	2494
41.4.12.2.7	Peripheral DMA Unit Copy Mode.....	2495
41.4.12.2.8	Error Management.....	2495
41.4.12.2.8.1	Immediate Errors.....	2495
41.4.12.2.8.2	Data Transfer Errors.....	2496
41.4.12.2.8.3	Read Error (First Phase).....	2496
41.4.12.2.8.4	Write Error and Read Error (Second Phase).....	2497
41.4.12.2.8.5	Copy Mode Errors.....	2498
41.4.12.2.8.6	Error Check Example.....	2498
41.4.12.2.9	Peripheral DMA Unit Prefetch/Flush Management.....	2499
41.4.12.3	OnCE and Real-Time Debug.....	2499
41.4.12.3.1	Memory and Register Access.....	2499
41.4.12.3.2	Hardware Breakpoints.....	2499
41.4.12.3.3	Watchpoints.....	2500
41.4.12.3.4	Software Breakpoints.....	2500
41.4.12.3.5	Core Control.....	2500

Section number	Title	Page
41.4.13	The OnCE Controller.....	2500
41.4.13.1	OnCE Commands.....	2500
41.4.13.2	Sending Commands to the OnCE Controller.....	2501
41.4.13.2.1	Using the JTAG Interface.....	2501
41.4.13.2.2	Using the ARM platform.....	2502
41.4.13.2.3	Conflicts Between the JTAG and the ARM platform Accesses.....	2503
41.4.13.3	Executing a Command from the OnCE.....	2504
41.4.13.3.1	Nature of the Commands.....	2504
41.4.13.3.2	Execution Request.....	2504
41.4.13.3.3	Command Execution.....	2505
41.4.13.4	Registers Descriptions.....	2507
41.4.13.4.1	Event Cell Counter Register (ECOUNT).....	2507
41.4.13.4.2	Event Cell Address Registers (EAA or EAB).....	2507
41.4.13.4.3	Event Cell Address Mask Register (EAM).....	2507
41.4.13.4.4	Event Cell Data Register (ED).....	2508
41.4.13.4.5	Event Cell Data Mask Register (EDM).....	2508
41.4.13.4.6	Real Time Buffer Register (RTB).....	2508
41.4.13.4.7	Event Control Register (ECTL).....	2508
41.4.13.4.8	Trace Buffer (TB).....	2509
41.4.13.4.9	OnCE Status Register (OSTAT).....	2509
41.4.13.5	JTAG Interface Requirements.....	2509
41.4.13.5.1	TCK Speed Limitation.....	2510
41.4.13.5.2	Synchronization Implementation.....	2510
41.4.13.5.3	JTAG Controller Start-Up Recommended Procedure.....	2512
41.4.14	Using the OnCE.....	2512
41.4.14.1	Activating Clocks in Debug Mode.....	2512
41.4.14.2	Getting the Current Status.....	2512
41.4.14.3	Methods of Entering Debug Mode.....	2513
41.4.14.3.1	External Debug Request During Reset.....	2513

Section number	Title	Page
41.4.14.3.2	Debug Request During Normal Activity.....	2513
41.4.14.3.3	Software Breakpoint Instruction.....	2513
41.4.14.3.4	Event Detection Unit Matching Condition.....	2513
41.4.14.4	Executing Instructions in Debug Mode.....	2514
41.4.14.5	Command Sequences Examples.....	2514
41.4.14.5.1	Getting the SDMA Status.....	2514
41.4.14.5.2	Saving the Context.....	2515
41.4.14.5.3	Restoring the Context.....	2516
41.4.14.5.4	Accessing the Memory.....	2517
41.4.14.5.5	Resuming Program Execution.....	2518
41.4.14.5.6	Single Stepping in RAM.....	2518
41.4.14.5.7	Single Stepping in ROM.....	2519
41.4.14.6	OnCE Event Detection Unit.....	2519
41.4.14.7	Clock Gating and Reset.....	2520
41.4.14.7.1	Clocks.....	2520
41.4.14.7.2	Resets.....	2521
41.4.14.8	Real Time Features.....	2521
41.4.14.8.1	Trace Buffer.....	2521
41.4.14.8.2	Real Time Buffer.....	2523
41.4.14.8.3	Emulation Pin.....	2523
41.4.14.8.4	Real-Time Debug Outputs.....	2523
41.5	Instruction Set.....	2527
41.5.1	Instruction Encoding.....	2527
41.5.2	SDMA Instruction Set.....	2529
41.5.2.1	ADD (Addition).....	2531
41.5.2.2	ADDI (Add with Immediate Value).....	2532
41.5.2.3	AND (Logical AND).....	2533
41.5.2.4	ANDI (Logical AND with Immediate Value).....	2534
41.5.2.5	ANDN (Logical AND NOT).....	2535

Section number	Title	Page
41.5.2.6	ANDNI (Logical AND with Negated Immediate Value).....	2536
41.5.2.7	ASR1 (Arithmetic Shift Right by 1 Bit).....	2537
41.5.2.8	BCLRI1 (Bit Clear Immediate).....	2538
41.5.2.9	BDF (Conditional Branch if Destination Fault).....	2539
41.5.2.10	BF (Conditional Branch if False).....	2540
41.5.2.11	BSETI (Bit Set Immediate).....	2541
41.5.2.12	BSF (Conditional Branch if Source Fault).....	2542
41.5.2.13	BT (Conditional Branch if True).....	2543
41.5.2.14	BTSTI (Bit Test immediate).....	2544
41.5.2.15	CLRF (Clear ARM platform flags).....	2545
41.5.2.16	CMPEQ (Compare for Equal).....	2546
41.5.2.17	CMPEQI (Compare with Immediate for Equal).....	2547
41.5.2.18	CMPHS (Compare for Higher or Same).....	2548
41.5.2.19	CMPLT (Compare for Less Than).....	2549
41.5.2.20	cpShReg (Update Context of PCU Registers and Flag).....	2550
41.5.2.21	DONE (DONE, Yield) .....	2550
41.5.2.22	ILLEGAL (ILLEGAL Instruction).....	2552
41.5.2.23	JMP (Unconditional Jump Immediate).....	2553
41.5.2.24	JMPR (Unconditional Jump).....	2553
41.5.2.25	JSR (Unconditional Jump to Subroutine Immediate).....	2554
41.5.2.26	JSRR (Unconditional Jump to Subroutine).....	2555
41.5.2.27	LD (Load Register).....	2556
41.5.2.28	LDF (Load Register from Functional Unit).....	2557
41.5.2.29	LDI (Load Register with Immediate Value).....	2559
41.5.2.30	LDRPC (Load from RPC to Register).....	2560
41.5.2.31	LOOP (Hardware Loop).....	2561
41.5.2.32	LSL1 (Logical Shift Left by 1 Bit).....	2563
41.5.2.33	LSR1 (Logical Shift Right by 1 Bit).....	2564
41.5.2.34	MOV (Logical Move).....	2565

Section number	Title	Page
41.5.2.35	NOTIFY (Notify to ARM platform).....	2566
41.5.2.36	OR (Logical OR).....	2567
41.5.2.37	ORI (Logical OR with Immediate Value).....	2568
41.5.2.38	RET (Return from Subroutine).....	2569
41.5.2.39	REVB (Reverse Byte Order).....	2570
41.5.2.40	Reverse Low Order Bytes(REVBLO).....	2570
41.5.2.41	ROR1 (Rotate Right by 1 Bit).....	2571
41.5.2.42	RORB (Rotate Right by 1 Byte).....	2572
41.5.2.43	SOFTBKPT (Software Breakpoint).....	2573
41.5.2.44	ST (Store Register).....	2573
41.5.2.45	STF (Store Register in Functional Unit).....	2575
41.5.2.46	SUB (Subtract).....	2578
41.5.2.47	SUBI (Subtract with Immediate).....	2579
41.5.2.48	TST (Test with Zero).....	2580
41.5.2.49	TSTI (Test Immediate).....	2581
41.5.2.50	XOR (Logical Exclusive OR).....	2582
41.5.2.51	XORI (Exclusive OR with Immediate).....	2583
41.5.2.52	YIELD, YIELDGE (DONE, Yield).....	2584
41.6	Software Restrictions.....	2584
41.6.1	Unsupported Burst DMA Access Sequence.....	2584
41.7	Application Notes.....	2585
41.7.1	Data Structures for Boot Code and Channel Scripts.....	2585
41.7.1.1	Buffer Descriptor Format.....	2586
41.7.1.2	Buffer Descriptor Commands for Bootload scripts.....	2589
41.7.1.3	Example of Buffer Descriptors for Channel 0.....	2590
41.7.1.4	Channel Context.....	2593
41.7.2	Typical Data Transfer Supported by SDMA DMA Units.....	2593
41.7.2.1	External Memory to External Memory.....	2594

Section number	Title	Page
41.7.2.2	Peripheral to Peripheral Transfer.....	2595
41.7.2.2.1	Source and Destination Target Have the Same Data Path Width.....	2595
41.7.2.2.2	Source and Destination Target Have a Different Data Path Width.....	2596
41.7.2.3	Transfer Between Peripheral and External Memory.....	2597
41.7.2.3.1	Peripheral to External Memory Transfer.....	2597
41.7.2.3.2	External Memory to Peripheral Transfer.....	2599
41.7.2.4	Transfer Between External Memory and Internal Memory.....	2600
41.7.2.4.1	Internal Memory to Internal Memory.....	2600
41.7.2.4.2	Transfer Between Peripheral and Internal Memory.....	2600
41.8	ARM Platform Memory Map and Control Register Definitions.....	2600
41.8.1	ARM platform Channel 0 Pointer (SDMAARM_MC0PTR).....	2606
41.8.2	Channel Interrupts (SDMAARM_INTR).....	2606
41.8.3	Channel Stop/Channel Status (SDMAARM_STOP_STAT).....	2606
41.8.4	Channel Start (SDMAARM_HSTART).....	2607
41.8.5	Channel Event Override (SDMAARM_EVTOVR).....	2607
41.8.6	Channel BP Override (SDMAARM_DSPOVR).....	2608
41.8.7	Channel ARM platform Override (SDMAARM_HOSTOVR).....	2608
41.8.8	Channel Event Pending (SDMAARM_EVTPEND).....	2608
41.8.9	Reset Register (SDMAARM_RESET).....	2609
41.8.10	DMA Request Error Register (SDMAARM_EVTERR).....	2610
41.8.11	Channel ARM platform Interrupt Mask (SDMAARM_INTRMASK).....	2610
41.8.12	Schedule Status (SDMAARM_PSW).....	2611
41.8.13	DMA Request Error Register (SDMAARM_EVTERRDBG).....	2611
41.8.14	Configuration Register (SDMAARM_CONFIG).....	2612
41.8.15	SDMA LOCK (SDMAARM_SDMA_LOCK).....	2613
41.8.16	OnCE Enable (SDMAARM_ONCE_ENB).....	2614
41.8.17	OnCE Data Register (SDMAARM_ONCE_DATA).....	2615
41.8.18	OnCE Instruction Register (SDMAARM_ONCE_INSTR).....	2615
41.8.19	OnCE Status Register (SDMAARM_ONCE_STAT).....	2615

Section number	Title	Page
41.8.20	OnCE Command Register (SDMAARM_ONCE_CMD).....	2617
41.8.21	Illegal Instruction Trap Address (SDMAARM_ILLINSTADDR).....	2618
41.8.22	Channel 0 Boot Address (SDMAARM_CHN0ADDR).....	2618
41.8.23	DMA Requests (SDMAARM_EVT_MIRROR).....	2619
41.8.24	DMA Requests 2 (SDMAARM_EVT_MIRROR2).....	2619
41.8.25	Cross-Trigger Events Configuration Register 1 (SDMAARM_XTRIG_CONF1).....	2620
41.8.26	Cross-Trigger Events Configuration Register 2 (SDMAARM_XTRIG_CONF2).....	2622
41.8.27	Channel Priority Registers (SDMAARM_SDMA_CHNPRI <sub>n</sub> ).....	2623
41.8.28	Channel Enable RAM (SDMAARM_CHNENBL <sub>n</sub> ).....	2623
41.9	BP Memory Map and Control Register Definitions.....	2624
41.9.1	Channel 0 Pointer (SDMABP_DC0PTR).....	2624
41.9.2	Channel Interrupts (SDMABP_INTR).....	2625
41.9.3	Channel Stop/Channel Status (SDMABP_STOP_STAT).....	2625
41.9.4	Channel Start (SDMABP_DSTART).....	2626
41.9.5	DMA Request Error Register (SDMABP_EVTERR).....	2626
41.9.6	Channel DSP Interrupt Mask (SDMABP_INTRMASK).....	2627
41.9.7	DMA Request Error Register (SDMABP_EVTERRDBG).....	2627
41.10	SDMA Internal (Core) Memory Map and Internal Register Definitions.....	2627
41.10.1	ARM platform Channel 0 Pointer (SDMACORE_MC0PTR).....	2629
41.10.2	Current Channel Pointer (SDMACORE_CCPTR).....	2629
41.10.3	Current Channel Register (SDMACORE_CCR).....	2629
41.10.4	Highest Pending Channel Register (SDMACORE_NCR).....	2630
41.10.5	External DMA Requests Mirror (SDMACORE_EVENTS).....	2631
41.10.6	Current Channel Priority (SDMACORE_CCPRI).....	2632
41.10.7	Next Channel Priority (SDMACORE_NCPRI).....	2632
41.10.8	OnCE Event Cell Counter (SDMACORE_ECOUNT).....	2633
41.10.9	OnCE Event Cell Control Register (SDMACORE_ECTL).....	2633
41.10.10	OnCE Event Address Register A (SDMACORE_EAA).....	2635
41.10.11	OnCE Event Cell Address Register B (SDMACORE_EAB).....	2635

Section number	Title	Page
41.10.12	OnCE Event Cell Address Mask (SDMACORE_EAM).....	2635
41.10.13	OnCE Event Cell Data Register (SDMACORE_ED).....	2636
41.10.14	OnCE Event Cell Data Mask (SDMACORE_EDM).....	2636
41.10.15	OnCE Real-Time Buffer (SDMACORE_RTB).....	2637
41.10.16	OnCE Trace Buffer (SDMACORE_TB).....	2637
41.10.17	OnCE Status (SDMACORE_OSTAT).....	2638
41.10.18	Channel 0 Boot Address (SDMACORE_MCHN0ADDR).....	2640
41.10.19	ENDIAN Status Register (SDMACORE_ENDIANNES).....	2641
41.10.20	Lock Status Register (SDMACORE_SDMA_LOCK).....	2642
41.10.21	External DMA Requests Mirror #2 (SDMACORE_EVENTS2).....	2643
41.11	SDMA Peripheral Registers.....	2643

## Chapter 42

### SiPix Display Controller (SPDC)

42.1	Overview.....	2645
42.1.1	Features.....	2645
42.1.2	Block Diagram.....	2646
42.2	External Signals.....	2646
42.2.1	Chip Level Connection.....	2647
42.2.1.1	Source/Gate Start Pulse (XDIO/YDIO) Connection.....	2647
42.2.1.2	Clock and Reset Signal Connection.....	2648
42.2.2	Chip Level Pinout List.....	2649
42.2.3	Display Function Timing Description.....	2651
42.2.3.1	Source Driver Timing.....	2651
42.2.3.2	Gate Driver Timing.....	2652
42.3	Software Programming Model.....	2653
42.3.1	SPDC Initial Setup Programming.....	2653
42.3.1.1	Working Memory and Bus Bandwidth Requirements.....	2654
42.3.1.1.1	SPDC Working Memory Buffer.....	2654
42.3.1.1.1.1	GRAY MODE.....	2654



Section number	Title	Page
42.3.1.1.1.2	COLOR MODE.....	2655
42.3.1.1.2	Bus Bandwidth Calculation.....	2656
42.3.1.2	Reading SiPix EPD Driving Waveform LUT.....	2656
42.3.1.3	SiPix EPD Panel Power Sequence Control.....	2657
42.3.1.4	SPDC System Clock Configuration.....	2657
42.3.1.5	Clock Gating Architecture Configuration.....	2659
42.3.1.6	Interrupts Configuration.....	2660
42.3.1.6.1	Interrupt Sources.....	2660
42.3.1.6.2	Enable and Disable Interrupts.....	2660
42.3.1.6.3	Handle and Clear Interrupts.....	2660
42.3.1.7	Initial Configuration Command Sequence.....	2660
42.3.2	Display Update Programming.....	2661
42.3.2.1	Basic Waveform Mode Introduction.....	2661
42.3.2.1.1	Mode 0 - Gray refresh mode with flashing.....	2662
42.3.2.1.2	Mode 1 - Gray refresh mode without flashing.....	2662
42.3.2.1.3	Mode 2 - Text mode.....	2663
42.3.2.1.4	Mode 4 - High speed handwriting mode.....	2664
42.3.2.2	Initiating a Display Flow in Normal and High Speed Handwriting Mode.....	2665
42.3.2.3	Concurrent Update.....	2666
42.4	SPDC Memory Map/Register Definition.....	2666
42.4.1	Display Trigger (SPDC_DISP_TRIGGER).....	2668
42.4.2	Display Coordinate (SPDC_UPDATE_X_Y).....	2669
42.4.3	Display Area Size (SPDC_UPDATE_W_H).....	2669
42.4.4	LUT Parameter Update (SPDC_LUT_PARA_UPDATE).....	2670
42.4.5	Display Normal Operation (SPDC_OPERATE).....	2670
42.4.6	SPDC Initial Setting (SPDC_PANEL_INIT_SET).....	2671
42.4.7	Environment Temperature (SPDC_TEMPER_SETTING).....	2674
42.4.8	Next Frame Memory Address (SPDC_NEXT_BUF).....	2675
42.4.9	Current Frame Memory Address (SPDC_CURRENT_BUF).....	2675

Section number	Title	Page
42.4.10	Previous Frame Memory Address (SPDC_PREVIOUS_BUFF).....	2676
42.4.11	Counter Frame Memory Address (SPDC_FRM_CNT_BUFF).....	2676
42.4.12	LUT Memory Address (SPDC_LUT_BUFF).....	2676
42.4.13	Interrupt Enable (SPDC_INT_EN).....	2677
42.4.14	Interrupt Status & Clear (SPDC_INT_ST_CLR).....	2678
42.4.15	SPDC Operation Status (SPDC_STATUS).....	2680
42.4.16	Panel Type Related Information (SPDC_PANEL_TYPE_VER).....	2681
42.4.17	SPDC IP Version (SPDC_TCON_VER).....	2682
42.4.18	All Clock Gating Enable (SPDC_SW_GATE_CLK).....	2683

## Chapter 43 System JTAG Controller (SJC)

43.1	Overview.....	2685
43.1.1	Features.....	2687
43.1.2	Modes of Operation.....	2687
43.2	External Signals.....	2689
43.2.1	External Signal Overview.....	2690
43.2.2	TAP Controller.....	2691
43.2.3	Accessing ExtraDebug Registers.....	2693
43.3	TAP Selection Block (TSB).....	2695
43.3.1	Select Mode Using Software.....	2696
43.4	Boundary Scan Register (BSR).....	2697
43.5	SoC JTAG Instruction Register (SJIR).....	2697
43.5.1	ID_CODE Instruction (IDCODE).....	2698
43.5.2	SAMPLE/PRELOAD Instruction.....	2699
43.5.3	EXTEST Instruction.....	2699
43.5.4	HIGHZ Instruction.....	2700
43.5.5	BYPASS Instruction.....	2700
43.5.6	ENABLE_ExtraDebug Instruction.....	2701
43.5.7	ENTER_DEBUG instruction.....	2701

Section number	Title	Page
43.5.8	TAP Select Instruction.....	2701
43.5.9	EXTEST_PULSE instruction.....	2702
43.5.10	EXTEST_TRAIN instruction.....	2702
43.6	Security.....	2702
43.6.1	JTAG Security Modes.....	2703
43.6.1.1	Mode 1: No Debug - Maximum Security.....	2703
43.6.1.2	Mode 2: Secure JTAG - High Security.....	2704
43.6.1.2.1	Challenge/Response Mechanism in System JTAG Mode.....	2704
43.6.1.3	Mode 3: JTAG Enabled - Low Security.....	2705
43.6.2	Software Enabled JTAG.....	2705
43.6.3	Kill Trace.....	2706
43.6.4	SJC Disable Fuse.....	2707
43.7	Functional Description.....	2708
43.7.1	Static Core Debug.....	2708
43.7.2	Reset Mechanism.....	2708
43.8	Initialization/Application Information.....	2709
43.9	SJC Memory Map/Register Definition.....	2710
43.9.1	General Purpose Unsecured Status Register 1 (SJC_GPUSR1).....	2711
43.9.2	General Purpose Unsecured Status Register 2 (SJC_GPUSR2).....	2713
43.9.3	General Purpose Unsecured Status Register 3 (SJC_GPUSR3).....	2713
43.9.4	General Purpose Secured Status Register (SJC_GPSSR).....	2714
43.9.5	Debug Control Register (SJC_DCR).....	2715
43.9.6	Security Status Register (SJC_SSR).....	2717
43.9.7	General Purpose Clocks Control Register (SJC_GPCCR).....	2720

## Chapter 44 Secure Non-Volatile Storage (SNVS)

44.1	SNVS overview.....	2721
44.1.1	SNVS features.....	2722
44.1.2	Modes of operation.....	2723

Section number	Title	Page
44.2	External Signals.....	2724
44.3	Clocks.....	2724
44.4	SNVS structure.....	2724
44.4.1	SNVS_HP (high power domain).....	2725
44.4.2	Non-secure real time counter.....	2726
44.4.2.1	Calibrating the time counter.....	2726
44.4.2.2	Time counter alarm.....	2727
44.4.2.3	Periodic interrupt.....	2727
44.5	SNVS_LP (low power domain).....	2727
44.5.1	Behavior during system power down.....	2728
44.5.2	Secure real time counter (SRTC).....	2728
44.5.2.1	Calibrating the SRTC time counter.....	2728
44.5.2.2	Time counter alarm (zmk).....	2728
44.5.3	Monotonic counter (MC).....	2729
44.6	SNVS reset and system power up.....	2730
44.6.1	PMIC Interface.....	2730
44.7	SNVS interrupts, alarms, and security violations.....	2732
44.8	Programming Guidelines.....	2732
44.8.1	RTC/SRTC control bits setting.....	2732
44.8.2	RTC/SRTC value read.....	2734
44.8.3	General initialization guidelines.....	2734
44.9	SNVS Memory Map/Register Definition.....	2734
44.9.1	SNVS_HP Lock Register (SNVS_HPLR).....	2737
44.9.2	SNVS_HP Command Register (SNVS_HPCOMR).....	2739
44.9.3	SNVS_HP Control Register (SNVS_HPCR).....	2741
44.9.4	SNVS_HP Status Register (SNVS_HPSR).....	2744
44.9.5	SNVS_HP Real Time Counter MSB Register (SNVS_HPRTCMR).....	2746
44.9.6	SNVS_HP Real Time Counter LSB Register (SNVS_HPRTCLR).....	2747
44.9.7	SNVS_HP Time Alarm MSB Register (SNVS_HPTAMR).....	2747

Section number	Title	Page
44.9.8	SNVS_HP Time Alarm LSB Register (SNVS_HPTALR).....	2748
44.9.9	SNVS_LP Lock Register (SNVS_LPLR).....	2749
44.9.10	SNVS_LP Control Register (SNVS_LPCR).....	2751
44.9.11	SNVS_LP Status Register (SNVS_LPSR).....	2754
44.9.12	SNVS_LP Secure Real Time Counter MSB Register (SNVS_LPSRTC MR).....	2756
44.9.13	SNVS_LP Secure Real Time Counter LSB Register (SNVS_LPSRTCLR).....	2756
44.9.14	SNVS_LP Time Alarm Register (SNVS_LPTAR).....	2757
44.9.15	SNVS_LP Secure Monotonic Counter MSB Register (SNVS_LPSMCMR).....	2757
44.9.16	SNVS_LP Secure Monotonic Counter LSB Register (SNVS_LPSMCLR).....	2758
44.9.17	SNVS_LP General Purpose Register (SNVS_LPGPR).....	2758
44.9.18	SNVS_HP Version ID Register 1 (SNVS_HPVIDR1).....	2759
44.9.19	SNVS_HP Version ID Register 2 (SNVS_HPVIDR2).....	2759

## Chapter 45

### Shared Peripheral Bus Arbiter (SPBA)

45.1	Overview.....	2761
45.1.1	Features.....	2762
45.1.2	Modes of operation.....	2763
45.2	Clocks.....	2763
45.3	Functional description.....	2764
45.3.1	Masters arbitration.....	2764
45.4	Resource ownership control.....	2767
45.4.1	Access control .....	2767
45.4.1.1	Peripheral access.....	2767
45.4.1.2	Peripheral Right Register access.....	2768
45.4.2	Owner election.....	2769
45.4.3	Ending ownership.....	2769
45.4.3.1	Software Controlled Ownership Ending.....	2769
45.4.4	The Un-owned State.....	2770

Section number	Title	Page
45.5	SPBA Memory Map/Register Definition.....	2770
45.5.1	Peripheral Rights Register (SPBA_PRR $n$ ).....	2772

## Chapter 46

### Sony/Philips Digital Interface (SPDIF)

46.1	Introduction .....	2775
46.1.1	Overview.....	2777
46.2	External Signals.....	2778
46.3	Clocks.....	2779
46.4	Functional Description.....	2779
46.4.1	SPDIF Receiver.....	2779
46.4.1.1	Audio Data Reception.....	2780
46.4.1.1.1	Application Note.....	2781
46.4.1.2	Channel Status Reception.....	2782
46.4.1.2.1	Channel Status Interrupt.....	2783
46.4.1.3	User Bit Reception.....	2783
46.4.1.4	Validity Flag Reception.....	2785
46.4.1.5	SPDIF Receiver Interrupt Exception Definition.....	2785
46.4.1.6	Standards Compliance.....	2786
46.4.1.7	SPDIF PLOCK Detection and Rxclk Output.....	2787
46.4.1.8	Measuring Frequency of SPDIF_RxClk.....	2787
46.4.2	SPDIF Transmitter.....	2787
46.4.2.1	Audio Data Transmission.....	2788
46.4.2.2	Channel Status Transmission.....	2789
46.4.2.3	Validity Flag Transmission.....	2789
46.5	SPDIF Memory Map/Register Definition.....	2789
46.5.1	SPDIF Configuration Register (SPDIF_SCR).....	2791
46.5.2	CDText Control Register (SPDIF_SRCD).....	2793
46.5.3	PhaseConfig Register (SPDIF_SRPC).....	2794
46.5.4	InterruptEn Register (SPDIF_SIE).....	2795

Section number	Title	Page
46.5.5	InterruptStat Register (SPDIF_SIS).....	2797
46.5.6	InterruptClear Register (SPDIF_SIC).....	2799
46.5.7	SPDIFRxLeft Register (SPDIF_SRL).....	2800
46.5.8	SPDIFRxRight Register (SPDIF_SRR).....	2801
46.5.9	SPDIFRxCChannel_h Register (SPDIF_SRC SH).....	2801
46.5.10	SPDIFRxCChannel_l Register (SPDIF_SRC SL).....	2802
46.5.11	UchannelRx Register (SPDIF_SR U).....	2802
46.5.12	QchannelRx Register (SPDIF_SR Q).....	2803
46.5.13	SPDIFTxLeft Register (SPDIF_ST L).....	2803
46.5.14	SPDIFTxRight Register (SPDIF_ST R).....	2804
46.5.15	SPDIFTxCChannelCons_h Register (SPDIF_ST C SCH).....	2804
46.5.16	SPDIFTxCChannelCons_l Register (SPDIF_ST C SCL).....	2805
46.5.17	FreqMeas Register (SPDIF_SRF M).....	2805
46.5.18	SPDIFTxCk Register (SPDIF_ST C).....	2806

## Chapter 47 System Reset Controller (SRC)

47.1	SRC Overview.....	2809
47.1.1	Features.....	2809
47.2	External Signals.....	2809
47.3	Clocks.....	2810
47.4	Top-level resets, power-up sequence and external supply integration.....	2811
47.4.1	Reset and Power-up Flow.....	2811
47.4.2	Finite-State Machine (FSM).....	2814
47.4.3	Power mode transitions.....	2815
47.5	Power-On Reset and power sequencing.....	2816
47.5.1	External POR using SRC_POR_B.....	2816
47.5.2	Internal POR.....	2817

Section number	Title	Page
47.6	Functional Description.....	2817
47.6.1	Reset Control.....	2817
47.6.1.1	Reset inputs and outputs.....	2817
47.6.1.2	Reset Handling.....	2819
47.6.1.2.1	Reset Qualification.....	2819
47.6.1.2.2	Reset Sequence and De-Assertion.....	2820
47.6.1.2.3	POR (SRC_POR_B).....	2820
47.6.1.2.4	COLD RESET.....	2821
47.6.1.2.5	WARM RESET.....	2822
47.6.2	Parallel Reset Requests.....	2823
47.6.3	Boot Mode Control.....	2824
47.6.3.1	BOOT_MODE Pin Latching.....	2824
47.7	SRC Memory Map/Register Definition.....	2825
47.7.1	SRC Control Register (SRC_SCR).....	2827
47.7.2	SRC Boot Mode Register 1 (SRC_SBMR1).....	2830
47.7.3	SRC Reset Status Register (SRC_SRSR).....	2830
47.7.4	SRC Interrupt Status Register (SRC_SISR).....	2833
47.7.5	SRC Interrupt Mask Register (SRC_SIMR).....	2835
47.7.6	SRC Boot Mode Register 2 (SRC_SBMR2).....	2837
47.7.7	SRC General Purpose Register 1 (SRC_GPR1).....	2838
47.7.8	SRC General Purpose Register 2 (SRC_GPR2).....	2838
47.7.9	SRC General Purpose Register 3 (SRC_GPR3).....	2839
47.7.10	SRC General Purpose Register 4 (SRC_GPR4).....	2839
47.7.11	SRC General Purpose Register 5 (SRC_GPR5).....	2839
47.7.12	SRC General Purpose Register 6 (SRC_GPR6).....	2840
47.7.13	SRC General Purpose Register 7 (SRC_GPR7).....	2840
47.7.14	SRC General Purpose Register 8 (SRC_GPR8).....	2840
47.7.15	SRC General Purpose Register 9 (SRC_GPR9).....	2841
47.7.16	SRC General Purpose Register 10 (SRC_GPR10).....	2841



Section number	Title	Page
<b>Chapter 48</b>		
<b>Synchronous Serial Interface (SSI)</b>		
48.1	Overview.....	2843
48.1.1	Features.....	2844
48.1.2	Modes of Operation.....	2845
48.2	External Signal Description.....	2845
48.2.1	Signals Overview.....	2845
48.3	Clocks.....	2849
48.4	SSI Transmit FIFO 0 & 1 Registers.....	2849
48.5	SSI Transmit Shift Register (TXSR).....	2850
48.6	SSI Receive FIFO 0 and 1 Registers.....	2852
48.7	SSI Receive Shift Register (RXSR).....	2853
48.8	Functional Description.....	2855
48.8.1	Operating Modes.....	2855
48.8.1.1	Normal Mode.....	2857
48.8.1.1.1	Normal Mode Transmit.....	2857
48.8.1.1.2	Normal Mode Receive.....	2858
48.8.1.2	Network Mode.....	2860
48.8.1.2.1	Network Mode Transmit.....	2861
48.8.1.2.2	Network Mode Receive.....	2862
48.8.1.3	Gated Clock Mode.....	2864
48.8.1.4	I2S Mode.....	2867
48.8.1.5	AC97 Mode.....	2869
48.8.1.5.1	AC97 Fixed Mode (SSI.SACNT[1]=0).....	2871
48.8.1.5.2	AC97 Variable Mode (SSI.SACNT[1]=1).....	2871
48.8.2	External Frame and Clock Operation.....	2872
48.8.2.1	Data Alignment Formats Supported.....	2872
48.8.3	SSI Architecture.....	2873

Section number	Title	Page
48.8.4	SSI Clocking.....	2874
48.8.4.1	SSI Clock and Frame Sync Generation.....	2875
48.8.4.2	DIV2, PSR and PM Bit Description.....	2876
48.8.5	Receive Interrupt Enable Bit Description.....	2878
48.8.6	Transmit Interrupt Enable Bit Description.....	2879
48.8.7	Internal Frame and Clock Shutdown.....	2880
48.8.8	Peripheral Bus Interface.....	2882
48.8.8.1	Transfer Lengths Supported.....	2882
48.8.8.2	Transfer Bus Errors.....	2882
48.8.8.3	Clock Rate.....	2883
48.8.9	Reset.....	2883
48.9	SSI Memory Map/Register Definition.....	2883
48.9.1	SSI Transmit Data Register n (SSIx_STXn).....	2886
48.9.2	SSI Receive Data Register n (SSIx_SRXn).....	2886
48.9.3	SSI Control Register (SSIx_SCR).....	2887
48.9.4	SSI Interrupt Status Register (SSIx_SISR).....	2889
48.9.5	SSI Interrupt Enable Register (SSIx_SIER).....	2895
48.9.6	SSI Transmit Configuration Register (SSIx_STCR).....	2899
48.9.7	SSI Receive Configuration Register (SSIx_SRCR).....	2901
48.9.8	SSI Transmit Clock Control Register (SSIx_STCCR).....	2903
48.9.9	SSI Receive Clock Control Register (SSIx_SRCCR).....	2905
48.9.10	SSI FIFO Control/Status Register (SSIx_SFCSR).....	2906
48.9.11	SSI AC97 Control Register (SSIx_SACNT).....	2910
48.9.12	SSI AC97 Command Address Register (SSIx_SACADD).....	2911
48.9.13	SSI AC97 Command Data Register (SSIx_SACDAT).....	2911
48.9.14	SSI AC97 Tag Register (SSIx_SATAG).....	2912
48.9.15	SSI Transmit Time Slot Mask Register (SSIx_STMSK).....	2912
48.9.16	SSI Receive Time Slot Mask Register (SSIx_SRMSK).....	2913
48.9.17	SSI AC97 Channel Status Register (SSIx_SACCST).....	2913

Section number	Title	Page
48.9.18	SSI AC97 Channel Enable Register (SSLx_SACCEN).....	2914
48.9.19	SSI AC97 Channel Disable Register (SSLx_SACCDIS).....	2914

## Chapter 49 Temperature Monitor (TEMPMON)

49.1	Overview.....	2915
49.2	Software Usage Guidelines.....	2916
49.3	TEMPMON Memory Map/Register Definition.....	2917
49.3.1	Tempsensor Control Register 0 (TEMPMON_TEMPSENSE0n).....	2918
49.3.2	Tempsensor Control Register 1 (TEMPMON_TEMPSENSE1n).....	2920

## Chapter 50 TrustZone Address Space Controller (TZASC)

50.1	Overview.....	2921
50.2	Clocks.....	2922
50.3	i.MX 6SoloLite Specific Configuration .....	2922
50.4	Address Mapping in various memory mapping modes.....	2923

## Chapter 51 Universal Asynchronous Receiver/Transmitter (UART)

51.1	Overview.....	2925
51.1.1	Features.....	2926
51.1.2	Modes of Operation.....	2927
51.2	External Signals.....	2927
51.2.1	Detailed Signal Descriptions.....	2929
51.2.1.1	Serial/IrDA Signals.....	2929
51.2.1.1.1	RXD - Data Receive.....	2929
51.2.1.1.2	TXD - Data Transmit.....	2929
51.2.1.2	Modem Control Signals.....	2929
51.2.1.2.1	CTS - Clear To Send .....	2929
51.2.1.2.2	RTS - Request To Send.....	2929
51.2.1.2.3	DSR - Data Set Ready.....	2930
51.2.1.2.4	DCD - Data Carrier Detected.....	2930

Section number	Title	Page
51.2.1.2.5	DTR - Data Terminal Ready.....	2930
51.2.1.2.6	RI - Ring Indicator.....	2930
51.2.1.3	Interrupt Signals.....	2930
51.2.1.3.1	interrupt_uart - UART Interrupt.....	2930
51.2.1.4	DMA Request Signals.....	2930
51.2.1.4.1	dma_req_rx - Receiver DMA Request.....	2930
51.2.1.4.2	dma_req_tx - Transmitter DMA Request.....	2931
51.2.1.5	Special Signals.....	2931
51.2.1.5.1	stop_req - Stop Mode.....	2931
51.2.1.5.2	doze_req - Doze Mode.....	2931
51.2.1.5.3	debug_req - Debug Mode.....	2931
51.3	Clocks.....	2931
51.4	Functional Description.....	2932
51.4.1	Interrupts and DMA Requests.....	2932
51.4.2	Clocks.....	2933
51.4.2.1	Clock requirements.....	2933
51.4.2.2	Maximum Baud Rate.....	2934
51.4.2.3	Clocking in Low-Power Modes.....	2934
51.4.3	General UART Definitions.....	2935
51.4.3.1	RTS_B - UART Request To Send.....	2936
51.4.3.2	RTS Edge Triggered Interrupt.....	2936
51.4.3.3	DTR_B - Data Terminal Ready .....	2937
51.4.3.4	DSR_B - Data Set Ready.....	2937
51.4.3.5	DTR_B/DSR_B Edge Triggered Interrupt.....	2938
51.4.3.6	DCD_B - Data Carrier Detect.....	2938
51.4.3.7	RI_B - Ring Indicator.....	2939
51.4.3.8	CTS_B - Clear To Send.....	2939
51.4.3.9	Programmable CTS_B Deassertion.....	2939
51.4.3.10	TX_DATA - UART Transmit.....	2939

Section number	Title	Page
51.4.3.11	RX_DATA - UART Receive.....	2940
51.4.4	Transmitter.....	2942
51.4.4.1	Transmitter FIFO Empty Interrupt Suppression.....	2942
51.4.4.2	Transmitting a Break Condition.....	2944
51.4.5	Receiver.....	2944
51.4.5.1	Idle Line Detect.....	2945
51.4.5.2	Aging Character Detect.....	2946
51.4.5.3	Receiver Wake.....	2947
51.4.5.4	Receiving a BREAK Condition.....	2948
51.4.5.5	Vote Logic.....	2948
51.4.5.6	Baud Rate Automatic Detection Logic.....	2950
51.4.5.6.1	Baud Rate Automatic Detection Protocol.....	2951
51.4.5.6.2	New Baud Rate Determination.....	2951
51.4.5.6.2.1	New Autobaud Counter Stopped bit and Interrupt.....	2952
51.4.6	Escape Sequence Detection.....	2952
51.5	Binary Rate Multiplier (BRM).....	2954
51.6	Infrared Interface.....	2956
51.6.1	Generalities-Infrared.....	2956
51.6.2	Inverted Transmission and Reception bits (INVT & INVR).....	2957
51.6.3	InfraRed Special Case (IRSC) Bit.....	2957
51.6.4	IrDA interrupt.....	2958
51.6.5	Conclusion about IrDA.....	2959
51.6.6	Programming IrDA Interface.....	2960
51.6.6.1	High Speed.....	2960
51.6.6.2	Low Speed.....	2960
51.7	9-bit RS-485 Mode.....	2961
51.7.1	Generalities.....	2961
51.7.2	Transmit 9-bit RS-485 frames.....	2962

Section number	Title	Page
51.7.3	Receive 9-bit RS-485 frames.....	2962
51.7.3.1	RS-485 Slave Address Normal Detect Mode.....	2962
51.7.3.2	RS-485 Slave Address Automatic Detect Mode.....	2963
51.8	Low Power Modes.....	2963
51.8.1	UART Operation in System Doze Mode.....	2964
51.8.2	UART Operation in System Stop Mode.....	2964
51.8.3	Power Saving Method in UART.....	2965
51.9	UART Operation in System Debug State.....	2965
51.10	Reset.....	2966
51.10.1	Hardware reset.....	2966
51.10.2	Software reset.....	2966
51.11	Transfer Error.....	2966
51.12	Functional Timing.....	2967
51.12.1	IrDA Mode.....	2967
51.13	Initialization.....	2967
51.13.1	Programming the UART in RS-232 mode.....	2967
51.13.2	Programming the UART in 9-bit RS-485 mode.....	2969
51.14	References.....	2970
51.15	UART Memory Map/Register Definition.....	2971
51.15.1	UART Receiver Register (UARTx_URXD).....	2976
51.15.2	UART Transmitter Register (UARTx_UTXD).....	2978
51.15.3	UART Control Register 1 (UARTx_UCR1).....	2979
51.15.4	UART Control Register 2 (UARTx_UCR2).....	2981
51.15.5	UART Control Register 3 (UARTx_UCR3).....	2984
51.15.6	UART Control Register 4 (UARTx_UCR4).....	2986
51.15.7	UART FIFO Control Register (UARTx_UFCR).....	2988
51.15.8	UART Status Register 1 (UARTx_USR1).....	2990
51.15.9	UART Status Register 2 (UARTx_USR2).....	2993
51.15.10	UART Escape Character Register (UARTx_UESC).....	2995

Section number	Title	Page
51.15.11	UART Escape Timer Register (UARTx_UTIM).....	2996
51.15.12	UART BRM Incremental Register (UARTx_UBIR).....	2996
51.15.13	UART BRM Modulator Register (UARTx_UBMR).....	2997
51.15.14	UART Baud Rate Count Register (UARTx_UBRC).....	2997
51.15.15	UART One Millisecond Register (UARTx_ONEMS).....	2998
51.15.16	UART Test Register (UARTx_UTS).....	2999
51.15.17	UART RS-485 Mode Control Register (UARTx_UMCR).....	3000

## Chapter 52

### Universal Serial Bus Controller (USB)

52.1	Overview.....	3003
52.1.1	Features.....	3004
52.1.2	Modes of Operation.....	3005
52.1.2.1	Normal Mode.....	3006
52.1.2.2	Low-Power Mode.....	3006
52.2	External Signals.....	3007
52.3	Functional Description.....	3008
52.3.1	USB 2.0 Controller Core 0.....	3008
52.3.1.1	Host Mode.....	3008
52.3.1.2	Peripheral (Device) Mode.....	3008
52.3.1.3	Pins Used for OTG 1 Controller.....	3009
52.3.2	USB 2.0 Controller Core 1.....	3009
52.3.2.1	Pins Used for OTG 2 Controller.....	3009
52.3.3	USB 2.0 Controller Core 2.....	3010
52.3.4	USB Power Control.....	3010
52.3.4.1	Entering Low Power Suspend Mode.....	3010
52.3.4.2	Wake-Up Events.....	3011
52.3.4.2.1	Host Mode Events.....	3011
52.3.5	Interrupts.....	3012
52.3.5.1	USB Core Interrupts.....	3012

Section number	Title	Page
52.3.5.2	USB Wake-Up Interrupts.....	3012
52.4	USB Operation Model.....	3012
52.4.1	Register Interface.....	3013
52.4.1.1	Configuration, Control and Status Register Set.....	3013
52.4.1.2	Identification Registers.....	3015
52.4.1.3	OTG Operations.....	3016
52.4.1.3.1	Register Bits.....	3016
52.4.2	Host Data Structures.....	3017
52.4.2.1	Periodic Frame List.....	3018
52.4.2.2	Asynchronous List Queue Head Pointer.....	3019
52.4.2.3	Isochronous (High-Speed) Transfer Descriptor (iTDD).....	3020
52.4.2.3.1	Next Link Pointer.....	3021
52.4.2.3.2	iTDD Transaction Status and Control List.....	3022
52.4.2.3.3	iTDD Buffer Page Pointer List (Plus).....	3023
52.4.2.4	Split Transaction Isochronous Transfer Descriptor (siTDD).....	3025
52.4.2.4.1	Next Link Pointer.....	3025
52.4.2.4.2	siTDD Endpoint Capabilities/Characteristics.....	3026
52.4.2.4.3	siTDD Transfer State.....	3027
52.4.2.4.4	siTDD Buffer Pointer List (plus).....	3028
52.4.2.4.5	siTDD Back Link Pointer.....	3029
52.4.2.5	Queue Element Transfer Descriptor (qTDD).....	3029
52.4.2.5.1	Next qTDD Pointer.....	3030
52.4.2.5.2	Alternate Next qTDD Pointer.....	3031
52.4.2.5.3	qTDD Token.....	3031
52.4.2.5.4	qTDD Buffer Page Pointer List.....	3035
52.4.2.6	Queue Head.....	3035
52.4.2.6.1	Queue Head Horizontal Link Pointer.....	3036
52.4.2.6.2	Queue Head Endpoint Capabilities/Characteristics.....	3037
52.4.2.6.3	Transfer Overlay-Queue Head.....	3039



Section number	Title	Page
52.4.2.7	Periodic Frame Span Traversal Node (FSTN).....	3040
52.4.2.7.1	FSTN Normal Path Pointer .....	3041
52.4.2.7.2	FSTN Back Path Link Pointer .....	3041
52.4.3	Host Operational Model .....	3042
52.4.3.1	Host Controller Initialization .....	3042
52.4.3.2	Port Routing and Control .....	3043
52.4.3.2.1	Port Routing Control through EHCI Configured (CF) Bit .....	3045
52.4.3.2.2	Port Routing Control through PortOwner and Disconnect Event .....	3046
52.4.3.2.3	Example Port Routing State Machine .....	3048
52.4.3.2.3.1	EHCI HC Owner .....	3048
52.4.3.2.3.2	Companion HC Owner .....	3049
52.4.3.2.4	Port Power .....	3049
52.4.3.2.5	Port Reporting Over-Current .....	3050
52.4.3.3	Suspend/Resume-Host Operational Model .....	3051
52.4.3.3.1	Port Suspend/Resume .....	3052
52.4.3.4	Schedule Traversal Rules .....	3054
52.4.3.4.1	Example - Preserving Micro-Frame Integrity .....	3056
52.4.3.4.1.1	Transaction Fit - A Best-Fit Approximation Algorithm .....	3057
52.4.3.5	Periodic Schedule Frame Boundaries vs Bus Frame Boundaries .....	3059
52.4.3.6	Periodic Schedule .....	3062
52.4.3.7	Managing Isochronous Transfers Using iTDs .....	3063
52.4.3.7.1	Host Controller Operational Model for iTDs .....	3064
52.4.3.7.2	Software Operational Model for iTDs .....	3066
52.4.3.7.2.1	Periodic scheduling threshold.....	3068
52.4.3.8	Asynchronous Schedule .....	3069
52.4.3.8.1	Adding Queue Heads to Asynchronous Schedule.....	3070
52.4.3.8.2	Removing Queue Heads from Asynchronous Schedule .....	3071
52.4.3.8.3	Empty Asynchronous Schedule Detection .....	3073

Section number	Title	Page
52.4.3.8.4	Restarting Asynchronous Schedule Before EOF .....	3074
52.4.3.8.4.1	Example Method for Restarting Asynchronous Schedule Traversal .....	3075
52.4.3.8.4.2	Async Sched Not Active .....	3076
52.4.3.8.4.3	Async Sched Active .....	3076
52.4.3.8.4.4	Async Sched Sleeping .....	3077
52.4.3.8.4.5	Example Derivation for AsyncSchedSleepTime.....	3077
52.4.3.8.5	Asynchronous schedule traversal: Start Event.....	3077
52.4.3.8.6	Reclamation Status Bit (USBSTS Register) .....	3078
52.4.3.9	Operational Model for Nak Counter.....	3078
52.4.3.9.1	Nak Count Reload Control .....	3080
52.4.3.9.1.1	Wait for List Head .....	3081
52.4.3.9.1.2	Do Reload .....	3081
52.4.3.9.1.3	Wait for Start Event .....	3081
52.4.3.10	Managing Control/Bulk/Interrupt Transfers through Queue Heads.....	3082
52.4.3.10.1	Fetch Queue Head .....	3084
52.4.3.10.2	Advance Queue .....	3085
52.4.3.10.3	Execute Transaction .....	3086
52.4.3.10.3.1	Interrupt Transfer Pre-condition Criteria .....	3086
52.4.3.10.3.2	Asynchronous Transfer Pre-operations and Pre-condition Criteria .....	3086
52.4.3.10.3.3	Transfer Type Independent Pre-operations.....	3086
52.4.3.10.3.4	Halting a Queue Head .....	3089
52.4.3.10.3.5	Asynchronous Schedule Park Mode .....	3090
52.4.3.10.4	Write Back qTD .....	3092
52.4.3.10.5	Follow Queue Head Horizontal Pointer .....	3092
52.4.3.10.6	Buffer Pointer List Use for Data Streaming with qTDs .....	3093
52.4.3.10.7	Adding Interrupt Queue Heads to the Periodic Schedule .....	3095
52.4.3.10.8	Managing Transfer Complete Interrupts from Queue Heads .....	3095

Section number	Title	Page
52.4.3.11	Ping Control.....	3096
52.4.3.12	Split Transactions .....	3097
52.4.3.12.1	Split Transactions for Asynchronous Transfers .....	3098
52.4.3.12.1.1	Asynchronous - Do Start Split.....	3099
52.4.3.12.1.2	Asynchronous - Do Complete Split .....	3099
52.4.3.12.2	Split Transaction Interrupt .....	3100
52.4.3.12.2.1	Split Transaction Scheduling Mechanisms for Interrupt .....	3101
52.4.3.12.2.2	Host Controller Operational Model for FSTNs.....	3104
52.4.3.12.2.3	Software Operational Model for FSTNs.....	3107
52.4.3.12.2.4	Tracking Split Transaction Progress for Interrupt Transfers ...	3108
52.4.3.12.2.5	Split Transaction Execution State Machine for Interrupt .....	3109
52.4.3.12.2.6	Rebalancing the periodic schedule .....	3115
52.4.3.12.3	Split Transaction Isochronous .....	3116
52.4.3.12.3.1	Split Transaction Scheduling Mechanisms for Isochronous ...	3116
52.4.3.12.3.2	Tracking Split Transaction Progress for Isochronous Transfers.....	3121
52.4.3.12.3.3	Split Transaction Execution State Machine for Isochronous .	3123
52.4.3.12.3.4	Periodic Isochronous - Do Start Split .....	3124
52.4.3.12.3.5	Periodic Isochronous - Do Complete Split .....	3126
52.4.3.12.3.6	Complete-Split for Scheduling Boundary Cases 2a, 2b .....	3129
52.4.3.12.3.7	Split Transaction for Isochronous - Processing Examples .....	3131
52.4.3.13	Host Controller Pause.....	3133
52.4.3.14	Port Test Modes -Host Operational Model.....	3134
52.4.3.15	Interrupts-Host Operational Model.....	3134
52.4.3.15.1	Transfer/Transaction Based Interrupts .....	3136
52.4.3.15.1.1	Transaction Error .....	3136
52.4.3.15.1.2	Serial Bus Babble.....	3137
52.4.3.15.1.3	Data Buffer Error .....	3137
52.4.3.15.1.4	USB Interrupt (Interrupt on Completion (IOC)) .....	3138

Section number	Title	Page
	52.4.3.15.1.5 Short Packet.....	3138
52.4.3.15.2	Host Controller Event Interrupts .....	3138
	52.4.3.15.2.1 Port Change Events .....	3139
	52.4.3.15.2.2 Frame List Rollover .....	3139
	52.4.3.15.2.3 Interrupt on Async Advance .....	3139
	52.4.3.15.2.4 Host System Error .....	3139
52.4.4	EHCI Deviation.....	3140
52.4.4.1	Embedded Transaction Translator Function.....	3141
	52.4.4.1.1 Capability Registers.....	3141
	52.4.4.1.2 Operational Registers.....	3142
	52.4.4.1.3 Discovery-EHCI Deviation.....	3142
	52.4.4.1.4 Data Structures.....	3142
	52.4.4.1.5 Operational Model.....	3143
	52.4.4.1.5.1 Micro- frame Pipeline.....	3143
	52.4.4.1.5.2 Split State Machines.....	3144
	52.4.4.1.5.3 Asynchronous Transaction Scheduling and Buffer Management.....	3145
	52.4.4.1.5.4 USB 2.0 - 11.17.3.....	3145
	52.4.4.1.5.5 USB 2.0 - 11.17.4.....	3145
	52.4.4.1.5.6 Periodic Transaction Scheduling and Buffer Management.....	3145
	52.4.4.1.5.7 USB 2.0 - 11.18.6.[1-2].....	3145
	52.4.4.1.5.8 USB 2.0 - 11.18.[7-8].....	3145
	52.4.4.1.5.9 Multiple Transaction Translators.....	3145
52.4.4.2	Device Operation.....	3146
52.4.4.3	USB_USBMODE Register.....	3146
	52.4.4.3.1 Non-Zero Fields the Register File.....	3146
	52.4.4.3.2 SOF Interrupt.....	3146
52.4.4.4	Embedded Design Interface.....	3146
	52.4.4.4.1 Frame Adjust Register.....	3147

Section number	Title	Page
52.4.4.5	Miscellaneous variations from EHCI.....	3147
52.4.4.5.1	Programmable Physical Interface Behaviour.....	3147
52.4.4.5.2	Discovery.....	3147
52.4.4.5.2.1	Port Reset.....	3147
52.4.4.5.2.2	Port Speed Detection.....	3148
52.4.4.5.3	Port Test Mode.....	3148
52.4.5	Device Data Structures.....	3148
52.4.5.1	Endpoint Queue Head (dQH).....	3149
52.4.5.1.1	Endpoint Capabilities/Characteristics.....	3150
52.4.5.1.2	Transfer Overlay-Endpoint Queue Head.....	3151
52.4.5.1.3	Current dTD Pointer.....	3151
52.4.5.1.4	Set-up Buffer.....	3152
52.4.5.2	Endpoint Transfer Descriptor (dTD).....	3152
52.4.6	Device Operational Model.....	3154
52.4.6.1	Device Controller Initialization.....	3154
52.4.6.2	Port State and Control.....	3156
52.4.6.2.1	Bus Reset.....	3158
52.4.6.2.2	Suspend/Resume.....	3159
52.4.6.2.2.1	Suspend.....	3159
52.4.6.2.2.2	Resume.....	3160
52.4.6.2.3	Managing Endpoints.....	3160
52.4.6.2.4	Endpoint Initialization.....	3161
52.4.6.2.5	Stalling.....	3162
52.4.6.2.6	Data Toggle .....	3162
52.4.6.2.6.1	Data Toggle Reset.....	3163
52.4.6.2.6.2	Data Toggle Inhibit.....	3163
52.4.6.2.6.3	Priming Transmit Endpoints.....	3163
52.4.6.2.6.4	Priming Receive Endpoints.....	3164

Section number	Title	Page
52.4.6.3	Operational Model For Packet Transfers.....	3164
52.4.6.3.1	Interrupt/Bulk Endpoint Operational Model.....	3165
52.4.6.3.1.1	Interrupt/Bulk Endpoint Bus Response Matrix.....	3166
52.4.6.3.2	Control Endpoint Operation Model.....	3167
52.4.6.3.2.1	Setup Phase.....	3167
52.4.6.3.2.2	Data Phase.....	3168
52.4.6.3.2.3	Status Phase.....	3169
52.4.6.3.2.4	Control Endpoint Bus Response Matrix.....	3169
52.4.6.3.3	Isochronous Endpoint Operational Model.....	3170
52.4.6.3.3.1	Isochronous Pipe Synchronization.....	3171
52.4.6.3.3.2	Isochronous Endpoint Bus Response Matrix.....	3172
52.4.6.4	Managing Queue Heads.....	3172
52.4.6.4.1	Queue Head Initialization.....	3173
52.4.6.4.2	Operational Model For Setup Transfers.....	3174
52.4.6.5	Managing Transfers with Transfer Descriptors.....	3175
52.4.6.5.1	Software Link Pointers.....	3175
52.4.6.5.2	Building a Transfer Descriptor.....	3175
52.4.6.5.3	Executing A Transfer Descriptor.....	3176
52.4.6.5.4	Transfer Completion.....	3177
52.4.6.5.5	Flushing/De-priming an Endpoint.....	3177
52.4.6.5.6	Device Error Matrix.....	3178
52.4.6.6	Servicing Interrupts.....	3179
52.4.6.6.1	High-Frequency Interrupts.....	3179
52.4.6.6.2	Low-Frequency Interrupts.....	3179
52.4.6.6.3	Error Interrupts.....	3179
52.5	USB Non-Core Memory Map/Register Definition.....	3180
52.5.1	USB OTG1 Control Register (USBNC_USB_OTG1_CTRL).....	3182
52.5.2	USB OTG2 Control Register (USBNC_USB_OTG2_CTRL).....	3184
52.5.3	USB Host Control Register (USBNC_USB_UH_CTRL).....	3186

Section number	Title	Page
52.5.4	USB Host HSIC Control Register (USBNC_USB_UH_HSIC_CTRL).....	3188
52.5.5	OTG1 UTMI PHY Control 0 Register (USBNC_USB_OTG1_PHY_CTRL_0).....	3189
52.5.6	OTG2 UTMI PHY Control 0 Register (USBNC_USB_OTG2_PHY_CTRL_0).....	3190
52.6	USB Core Memory Map/Register Definition.....	3191
52.6.1	Identification register (USBC_n_ID).....	3196
52.6.2	Hardware General (USBC_n_HWGENERAL).....	3197
52.6.3	Host Hardware Parameters (USBC_n_HWHOST).....	3198
52.6.4	Device Hardware Parameters (USBC_n_HWDEVICE).....	3199
52.6.5	TX Buffer Hardware Parameters (USBC_n_HWTXBUF).....	3199
52.6.6	RX Buffer Hardware Parameters (USBC_n_HWRXBUF).....	3200
52.6.7	General Purpose Timer #0 Load (USBC_n_GPTIMER0LD).....	3201
52.6.8	General Purpose Timer #0 Controller (USBC_n_GPTIMER0CTRL).....	3201
52.6.9	General Purpose Timer #1 Load (USBC_n_GPTIMER1LD).....	3203
52.6.10	General Purpose Timer #1 Controller (USBC_n_GPTIMER1CTRL).....	3203
52.6.11	System Bus Config (USBC_n_SBUSCFG).....	3204
52.6.12	Capability Registers Length (USBC_n_CAPLENGTH).....	3205
52.6.13	Host Controller Interface Version (USBC_n_HCIVERSION).....	3206
52.6.14	Host Controller Structural Parameters (USBC_n_HCSPARAMS).....	3206
52.6.15	Host Controller Capability Parameters (USBC_n_HCCPARAMS).....	3208
52.6.16	Device Controller Interface Version (USBC_n_DCIVERSION).....	3210
52.6.17	Device Controller Capability Parameters (USBC_n_DCCPARAMS).....	3210
52.6.18	USB Command Register (USBC_n_USBCMD).....	3212
52.6.19	USB Status Register (USBC_n_USBSTS).....	3216
52.6.20	Interrupt Enable Register (USBC_n_USBINTR).....	3220
52.6.21	USB Frame Index (USBC_n_FRINDEX).....	3222
52.6.22	Frame List Base Address (USBC_n_PERIODICLISTBASE).....	3223
52.6.23	Device Address (USBC_n_DEVICEADDR).....	3223
52.6.24	Next Asynch. Address (USBC_n_ASYNCLISTADDR).....	3224
52.6.25	Endpoint List Address (USBC_n_ENDPTLISTADDR).....	3225

Section number	Title	Page
52.6.26	Programmable Burst Size (USBC_n_BURSTSIZE).....	3226
52.6.27	TX FIFO Fill Tuning (USBC_n_TXFILLTUNING).....	3226
52.6.28	Endpoint NAK (USBC_n_ENDPTNAK).....	3228
52.6.29	Endpoint NAK Enable (USBC_n_ENDPTNAKEN).....	3228
52.6.30	Configure Flag Register (USBC_n_CONFIGFLAG).....	3229
52.6.31	Port Status & Control (USBC_n_PORTSC1).....	3230
52.6.32	On-The-Go Status & control (USBC_n_OTGSC).....	3236
52.6.33	USB Device Mode (USBC_n_USBMODE).....	3240
52.6.34	Endpoint Setup Status (USBC_n_ENDPTSETUPSTAT).....	3241
52.6.35	Endpoint Prime (USBC_n_ENDPTPRIME).....	3242
52.6.36	Endpoint Flush (USBC_n_ENDPTFLUSH).....	3243
52.6.37	Endpoint Status (USBC_n_ENDPTSTAT).....	3243
52.6.38	Endpoint Complete (USBC_n_ENDPTCOMPLETE).....	3244
52.6.39	Endpoint Control0 (USBC_n_ENDPTCTRL0).....	3245
52.6.40	Endpoint Control 1 (USBC_n_ENDPTCTRL1).....	3246
52.6.41	Endpoint Control 2 (USBC_n_ENDPTCTRL2).....	3249
52.6.42	Endpoint Control 3 (USBC_n_ENDPTCTRL3).....	3252
52.6.43	Endpoint Control 4 (USBC_n_ENDPTCTRL4).....	3254
52.6.44	Endpoint Control 5 (USBC_n_ENDPTCTRL5).....	3257
52.6.45	Endpoint Control 6 (USBC_n_ENDPTCTRL6).....	3260
52.6.46	Endpoint Control 7 (USBC_n_ENDPTCTRL7).....	3262

## Chapter 53

### Universal Serial Bus 2.0 Integrated PHY (USB-PHY)

53.1	USB PHY Overview.....	3267
53.2	Operation.....	3268
53.2.1	UTMI.....	3269
53.2.2	Digital Transmitter.....	3269
53.2.3	Digital Receiver.....	3269



Section number	Title	Page
53.2.4	Analog Receiver.....	3269
53.2.4.1	HS Differential Receiver.....	3270
53.2.4.2	Squelch Detector.....	3271
53.2.4.3	LS/FS Differential Receiver.....	3271
53.2.4.4	HS Disconnect Detector.....	3271
53.2.4.5	USB Plugged-In Detector.....	3271
53.2.4.6	Single-Ended USB_DP Receiver.....	3272
53.2.4.7	Single-Ended USB_DN Receiver.....	3272
53.2.4.8	9X Oversample Module.....	3272
53.2.5	Analog Transmitter.....	3272
53.2.5.1	Switchable High-Speed 45Ω Termination Resistors.....	3272
53.2.5.2	Low-Speed/Full-Speed Differential Driver.....	3273
53.2.5.3	High-Speed Differential Driver.....	3273
53.2.5.4	Switchable 1.5KΩ USB_DP Pullup Resistor.....	3273
53.2.5.5	Switchable 15KΩ USB_DP Pulldown Resistor.....	3273
53.2.6	Recommended Register Configuration for USB Certification.....	3275
53.2.7	Charger detection.....	3275
53.2.7.1	Charger detect control table.....	3275
53.2.7.2	Data pin contact detector.....	3275
53.2.7.3	Charger detector.....	3276
53.2.7.4	Charger detection software flow.....	3276
53.2.7.5	Dead Battery Protect.....	3278
53.3	USB PHY Memory Map/Register Definition.....	3278
53.3.1	USB PHY Power-Down Register (USBPHY <sub>x</sub> _PWD <sub>n</sub> ).....	3281
53.3.2	USB PHY Transmitter Control Register (USBPHY <sub>x</sub> _TX <sub>n</sub> ).....	3283
53.3.3	USB PHY Receiver Control Register (USBPHY <sub>x</sub> _RX <sub>n</sub> ).....	3284
53.3.4	USB PHY General Control Register (USBPHY <sub>x</sub> _CTRL <sub>n</sub> ).....	3286
53.3.5	USB PHY Status Register (USBPHY <sub>x</sub> _STATUS).....	3289
53.3.6	USB PHY Debug Register (USBPHY <sub>x</sub> _DEBUG <sub>n</sub> ).....	3291

Section number	Title	Page
53.3.7	UTMI Debug Status Register 0 (USBPHY <sub>x</sub> _DEBUG0_STATUS).....	3293
53.3.8	UTMI Debug Status Register 1 (USBPHY <sub>x</sub> _DEBUG1n).....	3294
53.3.9	UTMI RTL Version (USBPHY <sub>x</sub> _VERSION).....	3295
53.4	USB Analog Memory Map/Register Definition.....	3295
53.4.1	USB VBUS Detect Register (USB_ANALOG_USB1_VBUS_DETECTn).....	3297
53.4.2	USB Charger Detect Register (USB_ANALOG_USB1_CHRG_DETECTn).....	3298
53.4.3	USB VBUS Detect Status Register (USB_ANALOG_USB1_VBUS_DETECT_STAT).....	3300
53.4.4	USB Charger Detect Status Register (USB_ANALOG_USB1_CHRG_DETECT_STAT).....	3302
53.4.5	USB Misc Register (USB_ANALOG_USB1_MISCN).....	3303
53.4.6	USB VBUS Detect Register (USB_ANALOG_USB2_VBUS_DETECTn).....	3304
53.4.7	USB Charger Detect Register (USB_ANALOG_USB2_CHRG_DETECTn).....	3306
53.4.8	USB VBUS Detect Status Register (USB_ANALOG_USB2_VBUS_DETECT_STAT).....	3308
53.4.9	USB Charger Detect Status Register (USB_ANALOG_USB2_CHRG_DETECT_STAT).....	3310
53.4.10	USB Misc Register (USB_ANALOG_USB2_MISCN).....	3311
53.4.11	Chip Silicon Version (USB_ANALOG_DIGPROG).....	3312

## Chapter 54

### Ultra Secured Digital Host Controller (uSDHC)

54.1	Overview.....	3313
54.1.1	Features.....	3316
54.1.2	Modes and Operations.....	3317
54.1.2.1	Data transfer Modes.....	3317
54.2	External Signals.....	3317
54.2.1	Signals Overview.....	3322
54.3	Clocks.....	3323
54.4	Functional Description.....	3323
54.4.1	Data Buffer.....	3324
54.4.1.1	Write Operation Sequence.....	3326
54.4.1.2	Read Operation Sequence.....	3327
54.4.1.3	Data Buffer and Block Size.....	3327

Section number	Title	Page
54.4.1.4	Dividing Large Data Transfer.....	3328
54.4.1.5	External DMA Request.....	3329
54.4.2	DMA AHB Interface.....	3330
54.4.2.1	Internal DMA Request.....	3331
54.4.2.2	DMA Burst Length.....	3332
54.4.2.3	AHB Master Interface.....	3332
54.4.2.4	ADMA Engine.....	3332
54.4.2.4.1	ADMA Concept and Descriptor Format.....	3333
54.4.2.4.2	ADMA Interrupt.....	3334
54.4.2.4.3	ADMA Error.....	3335
54.4.3	Register Bank with IP Bus Interface.....	3335
54.4.3.1	SD Protocol Unit.....	3336
54.4.3.2	SD control misc.....	3337
54.4.3.3	SD Clock control.....	3337
54.4.3.4	Command control.....	3337
54.4.3.5	Data control.....	3338
54.4.4	Clock & Reset Manager.....	3338
54.4.5	Clock Generator.....	3339
54.4.6	SDIO Card Interrupt.....	3339
54.4.6.1	Interrupts in 1-bit Mode.....	3339
54.4.6.2	Interrupt in 4-bit Mode.....	3339
54.4.6.3	Card Interrupt Handling.....	3340
54.4.7	Card Insertion and Removal Detection.....	3341
54.4.8	Power Management and Wake Up Events.....	3342
54.4.8.1	Setting Wake Up Events.....	3343
54.4.9	MMC fast boot.....	3343
54.4.9.1	Boot operation.....	3343
54.4.9.2	Alternative boot operation.....	3344

Section number	Title	Page
54.5	Initialization/Application of uSDHC.....	3345
54.5.1	Command Send & Response Receive Basic Operation.....	3345
54.5.2	Card Identification Mode.....	3346
54.5.2.1	Card Detect.....	3346
54.5.2.2	Reset.....	3347
54.5.2.3	Voltage Validation.....	3348
54.5.2.4	Card Registry.....	3350
54.5.3	Card Access.....	3351
54.5.3.1	Block Write.....	3351
54.5.3.1.1	Normal Write.....	3351
54.5.3.1.2	DDR Write.....	3353
54.5.3.1.3	Write with Pause.....	3353
54.5.3.2	Block Read.....	3355
54.5.3.2.1	Normal Read.....	3355
54.5.3.2.2	DDR Read.....	3355
54.5.3.2.3	Read with Pause.....	3356
54.5.3.2.4	DLL (Delay Line) in Read Path.....	3357
54.5.3.3	Suspend Resume.....	3359
54.5.3.3.1	Suspend.....	3359
54.5.3.3.2	Resume.....	3360
54.5.3.4	ADMA Usage.....	3360
54.5.3.5	Transfer Error.....	3361
54.5.3.5.1	CRC Error.....	3361
54.5.3.5.2	Internal DMA Error.....	3361
54.5.3.5.3	Transfer ADMA Error.....	3362
54.5.3.5.4	Auto CMD12 Error.....	3362
54.5.3.6	Card Interrupt.....	3363
54.5.4	Switch Function.....	3363
54.5.4.1	Query, Enable and Disable SDIO High Speed Mode.....	3364

Section number	Title	Page
54.5.4.2	Query, Enable and Disable SD High Speed Mode/SDR50/SDR104/DDR50.....	3364
54.5.4.3	Query, Enable and Disable MMC High Speed Mode.....	3365
54.5.4.4	Set MMC Bus Width.....	3365
54.5.5	ADMA Operation.....	3365
54.5.5.1	ADMA1 Operation.....	3365
54.5.5.2	ADMA2 Operation.....	3366
54.5.6	Fast Boot Operation.....	3366
54.5.6.1	Normal fast boot flow .....	3367
54.5.6.2	Alternative fast boot flow.....	3367
54.5.6.3	Fast boot application case (in DMA mode).....	3368
54.6	Commands for MMC/SD/SDIO.....	3370
54.7	Software Restrictions.....	3375
54.7.1	Initialization Active.....	3375
54.7.2	Software Polling Procedure.....	3376
54.7.3	Suspend Operation.....	3376
54.7.4	Data Length Setting.....	3376
54.7.5	(A)DMA Address Setting.....	3376
54.7.6	Data Port Access.....	3377
54.7.7	Change Clock Frequency.....	3377
54.7.8	Multi-block Read.....	3377
54.8	uSDHC Memory Map/Register Definition.....	3377
54.8.1	DMA System Address (uSDHCx_DS_ADDR).....	3383
54.8.2	Block Attributes (uSDHCx_BLK_ATT).....	3384
54.8.3	Command Argument (uSDHCx_CMD_ARG).....	3385
54.8.4	Command Transfer Type (uSDHCx_CMD_XFR_TYP).....	3385
54.8.5	Command Response0 (uSDHCx_CMD_RSP0).....	3389
54.8.6	Command Response1 (uSDHCx_CMD_RSP1).....	3389
54.8.7	Command Response2 (uSDHCx_CMD_RSP2).....	3390
54.8.8	Command Response3 (uSDHCx_CMD_RSP3).....	3390

Section number	Title	Page
54.8.9	Data Buffer Access Port (uSDHCx_DATA_BUFF_ACC_PORT).....	3392
54.8.10	Present State (uSDHCx_PRES_STATE).....	3392
54.8.11	Protocol Control (uSDHCx_PROT_CTRL).....	3398
54.8.12	System Control (uSDHCx_SYS_CTRL).....	3403
54.8.13	Interrupt Status (uSDHCx_INT_STATUS).....	3406
54.8.14	Interrupt Status Enable (uSDHCx_INT_STATUS_EN).....	3412
54.8.15	Interrupt Signal Enable (uSDHCx_INT_SIGNAL_EN).....	3415
54.8.16	Auto CMD12 Error Status (uSDHCx_AUTOCMD12_ERR_STATUS).....	3418
54.8.17	Host Controller Capabilities (uSDHCx_HOST_CTRL_CAP).....	3421
54.8.18	Watermark Level (uSDHCx_WTMK_LVL).....	3424
54.8.19	Mixer Control (uSDHCx_MIX_CTRL).....	3425
54.8.20	Force Event (uSDHCx_FORCE_EVENT).....	3427
54.8.21	ADMA Error Status Register (uSDHCx_ADMA_ERR_STATUS).....	3430
54.8.22	ADMA System Address (uSDHCx_ADMA_SYS_ADDR).....	3432
54.8.23	DLL (Delay Line) Control (uSDHCx_DLL_CTRL).....	3433
54.8.24	DLL Status (uSDHCx_DLL_STATUS).....	3435
54.8.25	CLK Tuning Control and Status (uSDHCx_CLK_TUNE_CTRL_STATUS).....	3436
54.8.26	Vendor Specific Register (uSDHCx_VEND_SPEC).....	3438
54.8.27	MMC Boot Register (uSDHCx_MMC_BOOT).....	3441
54.8.28	Vendor Specific 2 Register (uSDHCx_VEND_SPEC2).....	3442
54.8.29	Tuning Control Register (uSDHCx_TUNING_CTRL).....	3444

## Chapter 55 Watchdog Timer (WDOG)

55.1	Overview.....	3447
55.1.1	Features.....	3449
55.2	External signals.....	3450
55.3	Clocks.....	3450

Section number	Title	Page
55.4	Functional description.....	3450
55.4.1	Timeout event.....	3451
55.4.1.1	Servicing WDOG to reload the counter.....	3451
55.4.2	Interrupt event .....	3452
55.4.3	Power-down counter event.....	3452
55.4.4	Low power modes.....	3452
55.4.4.1	STOP and DOZE mode.....	3452
55.4.4.2	WAIT mode.....	3453
55.4.5	Debug mode.....	3453
55.4.6	Operations.....	3453
55.4.6.1	Watchdog reset generation.....	3453
55.4.6.2	WDOG_B generation.....	3454
55.4.7	Reset.....	3456
55.4.8	Interrupt.....	3456
55.4.9	Flow Diagrams.....	3456
55.5	Initialization.....	3461
55.6	WDOG Memory Map/Register Definition.....	3462
55.6.1	Watchdog Control Register (WDOGx_WCR).....	3462
55.6.2	Watchdog Service Register (WDOGx_WSR).....	3464
55.6.3	Watchdog Reset Status Register (WDOGx_WRSR).....	3465
55.6.4	Watchdog Interrupt Control Register (WDOGx_WICR).....	3466
55.6.5	Watchdog Miscellaneous Control Register (WDOGx_WMCR).....	3467

## Chapter 56 Crystal Oscillator (XTALOSC)

56.1	Overview.....	3469
56.2	External Signals.....	3469
56.3	Crystal Oscillator 24 MHz.....	3470
56.3.1	Oscillator Configuration (24 MHz).....	3470
56.3.2	Bypass Configuration (24 MHz).....	3471

Section number	Title	Page
56.3.3	RC Oscillator (24 MHz).....	3472
56.3.4	Crystal Frequency Detection(24 MHz).....	3472
56.4	Crystal Oscillator 32 kHz.....	3472
56.4.1	Oscillator Configuration (32 kHz).....	3472
56.4.2	Bypass Configuration (32 kHz).....	3473
56.5	XTALOSC 24MHz Memory Map/Register Definition.....	3474
56.5.1	Miscellaneous Register 0 (XTALOSC24M_MISC0).....	3475
56.5.2	XTAL OSC MISC1 Control Register (XTALOSC24M_MISC1n).....	3479



# Chapter 1

## Introduction

### 1.1 About This Document

The i.MX 6SoloLite application processors are Freescale Semiconductor's latest additions to a growing family of multimedia-focused products offering high performance processing optimized for lowest power consumption.

The i.MX 6SoloLite processors feature Freescale's advanced implementation of the ARM<sup>®</sup>Cortex<sup>®</sup>-A9 core, which can be interfaced with DDR3-800, LV-DDR3-800 and LPDDR2-800 DRAM memory devices.

These products are suitable for applications such as:

- eReaders
- High-end Mobile Internet Devices and high-end PDAs
- Netbooks
- Nettops
- Portable navigation devices
- Gaming Consoles

#### 1.1.1 Audience

This manual is intended to be used by board-level product designers and product software developers. This manual assumes that the reader has a background in computer engineering and/or software engineering and understands concepts of digital system design, microprocessor architecture, Input / Output (I/O) devices, industry standard communication and device interface protocols.

## 1.1.2 Organization

This document covers the i.MX 6SoloLite at a system level and provides an architectural overview. Also covered are system memory map, system-level interrupt events, external pins and pin multiplexing, external memory, system debug, system boot, multimedia subsystem, power management, and system security.

## 1.1.3 Suggested Reading

This section lists additional reading materials that provide background for the information in this manual, as well as general information about the architecture.

### 1.1.3.1 General Information

The following documentation provides useful background information about the ARM Cortex-A9 processor.

For information about the ARM Cortex-A9 processor see:

- <http://infocenter.arm.com>

### 1.1.3.2 Related Documentation

Freescale documentation is available from the sources listed on the back cover of this manual; the document order numbers are included in parentheses for ease in ordering.

For a current list of documentation, refer to <http://www.freescale.com>.

## 1.1.4 Conventions

This document uses the following notational conventions:

#### **cleared / set**

When a bit takes the value zero, it is said to be cleared; when it takes a value of one, it is said to be set.

#### **mnemonics**

Instruction mnemonics are shown in lowercase bold

#### *italics*

Italics indicate variable command parameters, for example, **bcctrx**

Book titles in text are set in italics

15

An integer in decimal

0x

Prefix to denote hexadecimal number

0b

Prefix to denote binary number. Binary 0 and 1 are written without the prefix.

n'H4000CA00

n-bit Hexadecimal number

BLK\_REG\_NAME

Register names are all uppercase. The block mnemonic is prepended with an underscore delimiter (\_).

BLK\_REG[FIELD]

Fields within registers appear in brackets. For example, ESR[RLS] refers to the Receive Last Slot field of the ESAI Status Register.

BLK\_REG[ *n*]

Bit number *n* within register BLK.REG.

BLK\_REG[ *l:r*]

Register bit ranges. Ranges are indicated by the left-most bit number *l* and the right-most bit number *r* separated by a colon (:). For example, ESR[15:0] refers to the lower half word in the ESAI Status Register.

x, U

In some contexts, such as signal encodings, an unitalicized x indicates a don't care or uninitialized. The binary value could be 1 or 0.

*x*

An italicized *x* indicates an alphanumeric variable

*n, m*

Italicized *n* or *m* represent integer variables

!

Binary logic operator NOT

&amp;&amp;

Binary logic operator AND

||

Binary logic operator OR

^ or &lt;O+&gt;

Binary logic operator XOR

|

Bit-wise OR. For example, 0b0001 | 0b1000 yields the value 0b1001.

&amp;

Bit-wise AND. For example, 0b0001 & 0b1000 yields the value 0b0000.

{A,B}

Concatenation, where the  $n$ -bit value  $A$  is prepended to the  $m$ -bit value  $B$  to form an  $(n+m)$ -bit value. For example,  $\{0, \text{REG}_m[14:0]\}$  yields a 16-bit value with 0 in the most significant bit.

- or grey fill

Indicates a reserved bit field in an register. Although these bits can be written to as ones or zeros, they are always read as zeros.

>>

Shift right logical one position

<<

Shift left logical one position

= <left arrow>

Assignment

==

Compare equal

!=

Compare not equal

>

Greater than

<

Less than

## 1.1.5 Register Access

### 1.1.5.1 Register Diagram Field Access Type Legend

The following figure provides the interpretation of the notation used in register diagrams for a number of common field access types.

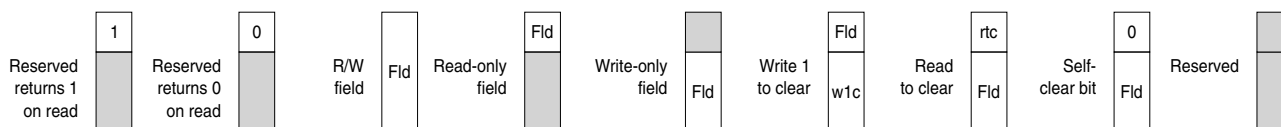


Figure 1-1. Register Field Conventions

#### NOTE

For reserved register fields, software should mask off the data in the field after read (software can not rely on the contents of data read from a reserved field) and always write all zeros.

### 1.1.5.2 Register Macro Usage

A common operation is to update one field without disturbing the contents of the remaining fields in the register. Normally, this requires a read-modify-write (RMW) operation, where the CPU reads the register, modifies the target field, then writes the results back to the register. This is an expensive operation in terms of CPU cycles, because of the initial register read.

To address this issue, some hardware registers are implemented as a group, including registers that can be used to either set, clear, or toggle (SCT) individual bits of the primary register. When writing to an SCT register, all bits set to 1 perform the associated operation on the primary register, while all bits set to 0 are not affected. The SCT registers always read back 0, and should be considered write-only. The SCT registers are not implemented if the primary register is read-only.

With this architecture, it is possible to update one or more fields using only register writes. First, all bits of the target fields are cleared by a write to the associated clear register, then the desired value of the target fields is written to the set register. This sequence of two writes is referred to as a clear-set (CS) operation.

A CS operation does have one potential drawback. Whenever a field is modified, the hardware sees a value of 0 before the final value is written. For most fields, passing through the 0 state is not a problem. Nonetheless, this behavior is something to consider when using a CS operation.

Also, a CS operation is not required for fields that are one bit wide. While the CS operation works in this case, it is more efficient to simply set or clear the target bit (that is, one write instead of two). A simple set or clear operation is also atomic, while a CS operation is not.

Note that not all macros for set, clear, or toggle (SCT) are atomic. For registers that do not provide hardware support for this functionality, these macros are implemented as a sequence of read/modify/write operations. When atomic operation is required, the developer should pay attention to this detail, because unexpected behavior might result if an interrupt occurs in the middle of the critical section comprising the update sequence.

### 1.1.6 Signal Conventions

**\_b, \_B**

When appended to a signal name, indicates that a signal is active-low

**NEG\_ACTIVE**

Overbar also denotes a negative active signal

**UPPERCASE**

Package pin names, Block I/O signals

**lowercase**

Lowercase is used to indicate internal signals

## 1.1.7 Acronyms and Abbreviations

The table below contains acronyms and abbreviations used in this document.

Acronyms and Abbreviated Terms

Term	Meaning
AHB	Advanced High-performance Bus
ALU	Arithmetic Logic Unit
AMBA	Advanced Microcontroller Bus Architecture
APB	Advanced Peripheral Bus
AXI	Advanced eXtensible Interface
BIST	Built-in self test
DDR	Double data rate
DMA	Direct memory access
DPLL	Digital phase-locked loop
DRAM	Dynamic random access memory
ECC	Error correcting codes
EPD	Electrophoretic Display
EPDC	Electrophoretic Display Controller
EPROM	Erasable programmable read-only memory
FIFO	First-in-first-out
GPIO	General-purpose I/O
GPR	General-purpose register
GPS	Global Positioning System
GPU	Graphics Processing Unit
HAB	High Assurance Boot
I2C or I <sup>2</sup> C	Inter-integrated circuit
IEEE	Institute of Electrical and Electronics Engineers
IrDA	Infrared Data Association
JTAG	Joint Test Action Group (a serial bus protocol usually used for test purposes)
LCD	Liquid Crystal Display
LCDIF	Liquid Crystal Display Interface
LDO	Low-Dropout
LIFO	Last-in-first-out
LRU	Least recently used
LSB	Least-significant byte
LUT	Lookup Table

*Table continues on the next page...*

Term	Meaning
LVDS	Low Voltage Differential Signaling
MAC	Medium Access Control
MMC	MultiMedia Card
MSB	Most-significant byte
MT/s	Mega transfers per second
OCRAM	On-Chip Random Access Memory
PCI	Peripheral Component Interconnect
PCle	PCI enhanced
PCMCIA	Personal Computer Memory Card International Association
PGC	Power Gating Controller
PIC	Programmable interrupt controller
POR	Power-on reset
PSRAM	Pseudo-Static Random Access Memory
QoS	Quality of Service
R2D	Radians to Degrees
RISC	Reduced instruction set computing
ROM	Read-Only Memory
RTOS	Real-time operating system
Rx	Receive
SCU	Snoop Control Unit
SD	Secure Digital
SDIO	Secure Digital Input/Output
SDLC	Synchronous data link control
SDMA	Smart DMA
SoC	System-On-Chip
SPDC	SiPix Display Controller
SPDIF	Sony Phillips Digital Interface
SPI	Serial peripheral interface
SRAM	Static random access memory
TFT	Thin Film Transistor
Tx	Transmit
UART	Universal asynchronous receiver/transmitter
USB	Universal serial bus
WLAN	Wireless Local Area Network
WXGA	Wide Extended Graphics Array

## 1.2 Introduction

This chapter introduces the architecture of the i.MX 6SoloLite Multimedia Applications Processor.

The i.MX 6SoloLite processor represents Freescale Semiconductor's latest achievement in integrated multimedia applications processors.

It is part of a growing family of multimedia-focused products, offering high performance processing optimized for the lowest power consumption.

## 1.3 Target Applications

The i.MX 6SoloLite applications processor is designed specifically for the eReader market. This device integrates the unique display controller for EPD panel, pixel processing engines, and graphics engines to make it an ideal solution for EPD-based eReaders. This device also has an integrated LCD controller that allows it to support eReaders with LCD panels.

The architecture's flexibility also allows for use in a wide variety of general embedded applications. The heart of the application chipset, the i.MX 6SoloLite processor provides all of the interfaces necessary for connecting peripherals such as WLAN, Bluetooth™, GPS, camera sensors, and multiple displays.

## 1.4 Features

The i.MX 6SoloLite processors are based on ARM Cortex-A9 MPCore™ Platform, which has the following features:

- Single ARM Cortex-A9 MPCore (with TrustZone) with:
  - 32 KB L1 Instruction Cache
  - 32 KB L1 Data Cache
  - Private Timer and Watchdog
  - Cortex-A9 NEON MPE (Media Processing Engine) Co-processor

The ARM Cortex-A9 MPCore complex includes:

- General Interrupt Controller (GIC) with 128 interrupt support
- Global Timer
- Snoop Control Unit (SCU)
- 256 KB unified I/D L2 cache:
- Two Master AXI (32-bit) bus interfaces output of L2 cache
- NEON MPE coprocessor



- SIMD Media Processing Architecture
- NEON register file with 32x64-bit general-purpose registers
- NEON Integer execute pipeline (ALU, Shift, MAC)
- NEON dual, single-precision floating point execute pipeline (FADD, FMUL)
- NEON load/store and permute pipeline
- Supports single and double-precision add, subtract, multiply, divide, multiply and accumulate, and square root operations as described in the ARM VFPv3 architecture.
- Provides conversions between 16-bit, 32-bit, and 64-bit floating-point formats and ARM integer word formats

Target frequency of the core (including Neon and L1 cache) is 800 MHz non-overdrive over the specified temperature range:

- 1 GHz overdrive over the specified temperature range
- 800 MHz non-overdrive over the specified temperature range

The i.MX 6SoloLite processors support latest, high volume, cost effective handheld DRAM, NOR Flash, and SD memory card standards.

The memory system consists of the following components:

- Level 1 Cache—32 KB Instruction, 32 KB Data cache per core
- Level 2 Cache—Unified instruction and data (256 KB)
- On-Chip Memory:
  - Boot ROM, including High Assurance Boot (HAB, 96 KB)
  - Internal multimedia / shared, fast access RAM (OCRAM, 128 KB)
- External memory interfaces:
  - 16/32-bit LP-DDR2, 16/32-bit DDR3-800, and LV-DDR3-800
  - 16-bit NOR Flash
  - 16-bit PSRAM, Cellular RAM

Each i.MX 6SoloLite processor enables the following interfaces to external devices (some of them are muxed and not available simultaneously):

Displays:

- EPDC supporting color and monochrome E-INK EPD panels up to 1650x2332 resolution and 5-bit grayscale
- SPDC supporting monochrome SiPix EPD panels up to 1600x1200 resolution
- LCDIF supporting one parallel 24-bit LCD display with up to WXGA(1366x768) resolution at 60 Hz
- Both LCD and a single EPD displays can be active simultaneously. This limitation exists because both EPD controllers are multiplexed over the same set of pins.

Camera sensors:

- Parallel Camera port (up to 16 bit and up to 66 MHz peak)

Expansion cards: Four MMC/SD/SDIO card ports all supporting:

- 1-bit or 4-bit transfer mode specifications for SD and SDIO cards up to UHS-I SDR-104 mode (104 MB/s max)
- 1-bit, 4-bit, or 8-bit transfer mode specifications for MMC cards up to 52 MHz in both SDR and DDR modes (104 MB/s max)

USB:

- Two High Speed (HS) USB 2.0 OTG (Up to 480 Mbps), with integrated HS USB Phy
- One High Speed (HS) USB 2.0 host with integrated HS-IC USB (High Speed Inter-Chip USB) Phy

Miscellaneous IPs and interfaces:

- Three I2S/SSI/AC97, up to 1.4 Mbps each
- Five UARTs operating up to 4.0 Mbps each, providing RS232 interface and supporting 9-bit RS485 multidrop mode. One UART supports 8-wire mode while the remaining four support 4-wire only.
- Four eCSPI (Enhanced CSPI), three of them can support up to 52 Mbps, one can be of low speed
- Three I2C, supporting 400 kbps
- Fast Ethernet Controller, 10/100 Mbps
- Four Pulse Width Modulators (PWM)
- System JTAG Controller (SJC)
- GPIO with interrupt capabilities
- 8x8 Key Pad Port (KPP)
- Sony Philips Digital Interface (SPDIF), Rx and Tx
- Two Watchdog timers (WDOG)
- Audio MUX (AUDMUX)

The i.MX 6SoloLite processors integrate advanced power management unit and controllers:

- Provide PMU, including multiple LDO supplies, for on-chip resources
- Use Temperature Sensor for monitoring the die temperature
- Support DVFS techniques for low power modes
- Use SW State Retention and Power Gating for ARM and MPE
- Support various levels of system power modes
- Use flexible clock gating control scheme

The i.MX 6SoloLite processors use dedicated HW accelerators to meet the targeted multimedia performance. The use of HW accelerators is a key factor in obtaining high performance at low power consumption numbers, while having the CPU core relatively free for performing other tasks.

The i.MX 6SoloLite processors incorporate the following hardware accelerators:

- GPU2Dv2—2D Graphics Processing Unit (BitBlt)
- GPUVG—OpenVG 1.1 Graphics Processing Unit
- PXP—PiXeL Processing Pipeline. Off loading key pixel processing operations are required to support the EPD display applications.

Security functions are enabled and accelerated by the following hardware:

- ARM TrustZone including the TZ architecture (separation of interrupts, memory mapping, etc.)
- SJC—System JTAG Controller. Protecting JTAG from debug port attacks by regulating or blocking the access to the system debug features.
- DCP—Data Co-Processor for cryptographic function, with 128-bit AES for Bulk Encryption, ECB and CCB modes, supports SHA-1, SHA-256 for Message Authentication/Integrity.
- RNGB—True Random Number Generation.
- SNVS—Secure Non-Volatile Storage, including Secure Real Time Clock.
- CSU—Central Security Unit. Enhancement for the IC Identification Module (IIM). Will be configured during boot and by eFUSES and will determine the security level operation mode as well as the TZ policy.
- HAB—High Assurance Boot (Secure Boot) v4.1.

### NOTE

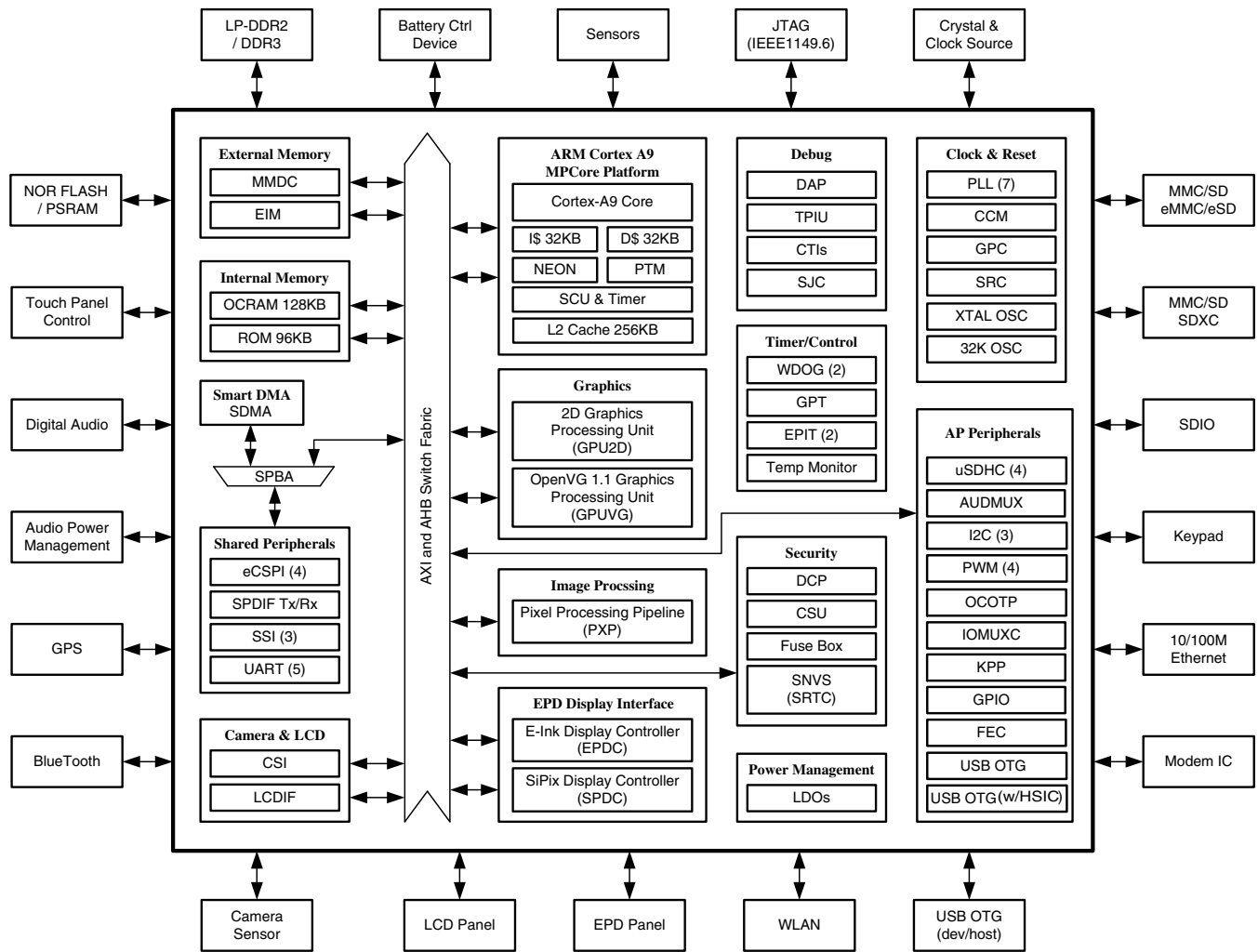
The actual feature set depends on the part number. Functions, such as 2D and OpenVG hardware graphics acceleration may not be enabled for specific part numbers.

## 1.5 Architectural Overview

This section contains i.MX 6SoloLite architectural details.

### 1.5.1 Simplified Block Diagram

A high level block diagram is shown in the figure below. This diagram provides a view of the i.MX 6SoloLite major sub-systems (processor domains, shared peripherals domain, memories, and so on) and logical connectivity.



**Figure 1-2. Simplified Block Diagram**

## 1.5.2 Architectural Partitioning

Architecture supports processing-intensive tasks in the following ways:

- ARM Cortex A9 MPCore™ Platform is responsible for:
  - Operating System
  - User applications (including control over hardware accelerators and non-accelerated functions)
  - TrustZone applications
  - 256KB L2 Cache memory inside ARM platform can be configured into OCRAM mode and used as normal OCRAM
- Smart DMA enables data transfer between non-mastering peripherals and external or internal memories
- System Control is supported via:

- Clock Control Module (CCM)
- Seven PLLs
- XTALOSC- 24MHz Crystal oscillator source support
- OSC32KHz - 32.768Hz Crystal oscillator source support
- System Reset Controller (SRC)
- General Power Controller (GPC)
- Temperature Sensor for monitoring and alarming on high temperature situations.
- Multimedia is supported with:
  - Graphics Processing Unit (GPU2Dv2)
    - Full function programmable 2D pipeline
    - Supports multiple 1080p surfaces
    - Programmable 9x9 tap filter
    - Range of image formats video and image manipulation
  - Vector graphics processing unit (GPUVGv2):
    - OpenVG 1.1 support
    - Full fixed function hardware vector graphics GPU
    - Hardware Tessellation (Minimum CPU involvement)
    - 16x FSAA, Photorealistic quality, No performance degradation
    - Multi-format rendering, including sRGB color transformation and video image conversion — High-quality vector font rendering support
    - Dedicated GPU for QoS requirements
  - Audio
    - Audio codecs are provided by SW, which runs on ARM core, supporting (but not limited to) MP3, WMA, AAC, HE-AAC and Pro10
    - 3x SSIs interfaced to Audio Mux
    - SPDIF Tx/Rx
  - Video
    - Video codecs are provided by SW, which runs on ARM core, supporting (but not limited to) MPEG4 and H.264.
- Security is supported by:
  - High Assurance Boot (HAB4) System
  - ARM TrustZone (TZ) Trusted Execution environment
  - DDR Memory secure region protection by TrustZone Address Space Controller
  - On-chip RAM (OCRAM) single region TrustZone protection
  - Peripheral access policy control, using Central Security Unit (CSU)
  - One-Time Programmable (OTP) electrical fuse array (Total of 2048-bit e-fuses) via the On-chip electrical fuse controller (OCOTP\_CTRL)
  - DCP and SNVS security architecture, providing:
    - Secure Real Time Counters (SRTC)
    - Security State Controller

- Random Number Generator (RNGB)
- Security Violation/Tamper Detection & Reporting
- System JTAG controller (SJC)
- TrustZone Watchdog (TZ WDOG)
- Connectivity peripherals, timers and External Memory Interfaces:
  - Embedded DMAs
  - Configurable 1.8V/3.3V IO voltage for seamless integration
  - Three USB 2.0 ports, including PHYs: 2x HS-USB OTG and 1x HS-IC USB HOST integrated PHYs
  - DDR2 Memory: 800 Mbps/line
  - LP-DDR2/DDR3/LVDDR3 (32-bit or 16-bit)
  - Timers: 2xEPIT, GPT and two Watch Dog timers (one of which is used for TZ), in addition to the timers and watchdog timers integrated within the ARM Cortex A9 MPCore™ platform.
  - Miscellaneous connectivity support -MMC/SD, I2C, SPI, UART, PWM and Keypad interface

### 1.5.3 Endianness Support

i.MX 6SoloLite supports Little Endian mode only.

### 1.5.4 Memory Interfaces

i.MX 6SoloLite supports the following memory interfaces:

- LP-DDR2, 32-bit, 400MHz clock
- DDR3 / LV-DDR3 32-bit, 400MHz clock.

NOR Flash, SRAM and PSRAM, 16/8-bit NOR Flash, interface is supported by the EIM block. All EIM pins are muxed on other interfaces.

## Chapter 2

# Memory Maps

## 2.1 Memory system overview

This chapter introduces the memory architecture of the i.MX 6SoloLite device.

## 2.2 ARM platform memory map

The i.MX 6SoloLite memory map has been provided in the following tables. The mapping of the DDR memory address space can be configured by software or hardware fuses, depending on the memory device type (LPDDR2/DDR2/DDR3) and the selected interleave/non-interleaving scheme.

**Table 2-1. System memory map**

Start address	End address	Size	Description
8000_0000	FFFF_FFFF	2048 MB	MMDC - DDR Controller.
1000_0000	7FFF_FFFF	1792 MB	Reserved
0800_0000	0FFF_FFFF	128 MB	EIM - Memory
02C0_0000	07FF_FFFF	84 MB	Reserved
02A0_0000	02DF_FFFF	4 MB	Reserved
0260_0000	029F_FFFF	4 MB	
0220_C000	023F_FFFF	2 MB	Reserved
0220_8000	0220_BFFF	16 KB	Reserved
0220_4000	0220_7FFF	16 KB	OpenVG (GC355)
0220_0000	0220_3FFF	16 KB	GPU2D (GC320)
0210_0000	021F_FFFF	1 MB	Peripheral IPs via AIPS-2 - See the AIPS map in <a href="#">Table 2-3</a> .
0200_0000	020F_FFFF	1 MB	Peripheral IPs via AIPS-1 - See the AIPS map in <a href="#">Table 2-2</a> .
01FF_C000	01FF_FFFF	16 KB	Reserved

*Table continues on the next page...*

**Table 2-1. System memory map (continued)**

Start address	End address	Size	Description
0100_0000	01FF_BFFF	16368 KB	Reserved
00E0_0000	00FF_FFFF	2048 KB	Reserved
00D0_0000	00DF_FFFF	1 MB	GPV_2 PL301 configuration port
00C0_0000	00CF_FFFF	1 MB	GPV_1 PL301 configuration port
00B0_0000	00BF_FFFF	1 MB	GPV_0 PL301 configuration port
00A0_3000	00AF_FFFF	1012 KB	Reserved
00A0_2000	00A0_2FFF	4 KB	PL310 (L2 Cache controller)
00A0_0000	00A0_1FFF	8 KB	ARM MP 0000 - 00FCh SCU registers 0100 - 01FFh Interrupt controller interfaces 0200 - 02FFh Global timer 0300 - 05FFh Reserved 0600 - 06FFh Private timers and watchdogs 0700 - 0FFFh Reserved 1000 - 1FFFh Interrupt distributor
009C_0000	009F_FFFF	256 KB	L2 Cache memory used as OCRAM aliased
0098_0000	009B_FFFF	256 KB	L2 Cache memory used as OCRAM 256KB
0092_0000	0097_FFFF	384 KB	OCRAM aliased
0090_0000	0091_FFFF	128 KB	OCRAM128 KB
0080_0000	008F_FFFF	1 MB	Reserved
0040_0000	007F_FFFF	4 MB	Reserved
0030_0000	003F_FFFF	1 MB	Reserved
0020_0000	002F_FFFF	1 MB	Reserved
0013_C000	001F_FFFF	784 KB	Reserved
0013_8000	0013_BFFF	16 KB	Reserved
0013_4000	0013_7FFF	16 KB	Reserved
0013_0000	0013_3FFF	16 KB	Reserved
0012_9000	0012_FFFF	28 KB	Reserved
0012_0000	0012_8FFF	36 KB	Reserved
0011_8000	0011_FFFF	32 KB	Reserved
0011_4000	0011_7FFF	16 KB	Reserved
0011_2000	0011_3FFF	8 KB	Reserved
0011_0000	0011_1FFF	8 KB	Reserved
0010_4000	0010_FFFF	48 KB	Reserved
0010_0000	0010_3FFF	16 KB	Reserved
0001_8000	000F_FFFF	928 KB	Reserved
0000_0000	0001_7FFF	96 KB	Boot ROM (ROMCP)

Table 2-2 shows the AIPS-1 detailed memory map.



**Table 2-2. AIPS-1 memory map**

Start Address	End Address	Region	NIC Port	Size
020F_C000	020F_FFFF	AIPS-1	DCP	16 KB
020F_8000	020F_BFFF		LCDIF	16 KB
020F_4000	020F_7FFF		EPDC	16 KB
020F_0000	020F_3FFF		PXP	16 KB
020E_C000	020E_FFFF		SDMA	16 KB
020E_8000	020E_BFFF		SPDC	16 KB
020E_4000	020E_7FFF		CSI	16 KB
020E_0000	020E_3FFF		IOMUXC	16 KB
020D_C2C0	020D_FFFF		Reserved	15680 B
020D_CA00	020D_C2BF		PGC_ARM	32 B
020D_C280	020D_C27F		Reserved	32 B
020D_C260	020D_C27F		PGC_PU	32 B
020D_C000	020D_C25F		GPC	16 KB
020D_8000	020D_BFFF		SRC	16 KB
020D_4000	020D_7FFF		EPIT2	16 KB
020D_0000	020D_3FFF		EPIT1	16 KB
020C_C000	020C_FFFF		SNVS_HP	16 KB
020C_B000	020C_BFFF		Reserved	4 KB
020C_A000	020C_AFFF		USBPHY2	4 KB
020C_9000	020C_9FFF		USBPHY1	4 KB
020C_8000	020C_8FFF		ANALOG: (PLLs, PFDs, Regulators, LDOs, Temp Sensor)	4 KB

*Table continues on the next page...*

**Table 2-2. AIPS-1 memory map (continued)**

Start Address	End Address	Region	NIC Port	Size
020C_4000	020C_7FFF	AIPS-1	CCM	16 KB
020C_0000	020C_3FFF		WDOG2	16 KB
020B_C000	020B_FFFF		WDOG1	16 KB
020B_8000	020B_BFFF		KPP	16 KB
020B_4000	020B_7FFF		Reserved	16 KB
020B_0000	020B_3FFF		Reserved	16 KB
020A_C000	020A_FFFF		GPIO5	16 KB
020A_8000	020A_BFFF		GPIO4	16 KB
020A_4000	020A_7FFF		GPIO3	16 KB
020A_0000	020A_3FFF		GPIO2	16 KB
0209_C000	0209_FFFF		GPIO1	16 KB
0209_8000	0209_BFFF		GPT	16 KB
0209_4000	0209_7FFF		Reserved	16 KB
0209_0000	0209_3FFF		DBGMON	16 KB
0208_C000	0208_FFFF		PWM4	16 KB
0208_8000	0208_BFFF		PWM3	16 KB
0208_4000	0208_7FFF		PWM2	16 KB
0208_0000	0208_3FFF		PWM1	16 KB
0207_C000	0207_FFFF		AIPS-1 Configuration	16 KB
0204_0000	0207_BFFF	AIPS-1 Global Module enable	Reserved	240 KB
0203_C000	0203_FFFF	AIPS-1 (via SPBA) Global Module Enable	SPBA	16 KB
0203_8000	0203_BFFF		UART4	16 KB
0203_4000	0203_7FFF		UART3	16 KB
0203_0000	0203_3FFF		SSI3	16 KB
0202_C000	0202_FFFF		SSI2	16 KB
0202_8000	0202_BFFF		SSI1	16 KB
0202_4000	0202_7FFF		UART2	16 KB
0202_0000	0202_3FFF		UART1	16 KB
0201_C000	0201_FFFF		Reserved for SDMA internal registers	16 KB
0201_8000	0201_BFFF		UART5	16 KB
0201_4000	0201_7FFF		eCSPI4	16 KB
0201_0000	0201_3FFF		eCSPI3	16 KB
0200_C000	0200_FFFF		eCSPI2	16KB
0200_8000	0200_BFFF		eCSPI1	16 KB
0200_4000	0200_7FFF		SPDIF	16 KB
0200_0000	0200_3FFF		Reserved for SDMA internal registers	16 KB

Table 2-3 shows the AIPS-2 detailed memory map.

**Table 2-3. AIPS-2 memory map**

Start Address	End Address	Region	Allocation	Size
021F_C000	021F_FFFF	AIPS-2	Reserved	16 KB
021F_8000	021F_BFFF		Reserved	16 KB
021F_4000	021F_7FFF		Reserved	16 KB
021F_0000	021F_3FFF		Reserved	16 KB
021E_C000	021E_FFFF		Reserved	16 KB
021E_8000	021E_BFFF		Reserved	16 KB
021E_4000	021E_7FFF		Reserved	16 KB
021E_0000	021E_3FFF		Reserved	16 KB
021D_C000	021D_FFFF		Reserved	16 KB
021D_8000	021D_BFFF		AUDMUX	16 KB
021D_4000	021D_7FFF		Reserved	16 KB
021D_0000	021D_3FFF		TZASC1	16 KB
021C_C000	021C_FFFF		Reserved	16 KB
021C_8000	021C_BFFF		Reserved	16 KB
021C_4000	021C_7FFF		Reserved	16 KB
021C_0000	021C_3FFF		CSU	16 KB
021B_C000	021B_FFFF		OCOTP_CTRL (wrapper)	16 KB
021B_8000	021B_BFFF		EIM - Registers	16 KB
021B_4000	021B_7FFF		RNGB	16 KB

*Table continues on the next page...*

**Table 2-3. AIPS-2 memory map (continued)**

Start Address	End Address	Region	Allocation	Size
021B_0000	021B_3FFF	AIPS-2	MMDC (port 0)	16 KB
021A_C000	021A_FFFF		ROMCP	16 KB
021A_8000	021A_BFFF		I2C3	16 KB
021A_4000	021A_7FFF		I2C2	16 KB
021A_0000	021A_3FFF		I2C1	16 KB
0219_C000	0219_FFFF		uSDHC4	16 KB
0219_8000	0219_BFFF		uSDHC3	16 KB
0219_4000	0219_7FFF		uSDHC2	16 KB
0219_0000	0219_3FFF		uSDHC1	16 KB
0218_C000	0218_FFFF		Reserved	16 KB
0218_8000	0218_BFFF		FEC	16 KB
0218_4000	0218_7FFF		USBO2H (USB)	16 KB
0218_0000	0218_3FFF		USBO2H (pl301)	16 KB
0217_C000	0217_FFFF		AIPS-2 configuration	16 KB
0216_1000	0217_BFFF		ARM Cortex A9 MPCore Platform - Reserved	108 KB
0214_0000	0216_0FFF		ARM Cortex A9 MPCore Platform / DAP	132 KB (See <a href="#">Table 2-4</a> )
0211_0000	0213_FFFF		Reserved	192 KB
0210_0000	0210_FFFF		Reserved	64 KB

[Table 2-4](#) shows the DAP detailed memory map.

**Table 2-4. DAP memory map**

Start Address	End Address	Region	NIC Port	Size	Allocation
0216_0000	0216_0FFF	ARM Cortex A9 MPCore Platform, DAP	AIPS-2, Global Map	4 KB	Platform Control
0215_F000	0215_FFFF			4 KB	Reserved
0215_E000	0215_EFFF			4 KB	Reserved
0215_D000	0215_DFFF			4 KB	Reserved
0215_C000	0215_CFFF			4 KB	PTM
0215_B000	0215_BFFF			4 KB	Reserved
0215_A000	0215_AFFF			4 KB	Reserved
0215_9000	0215_9FFF			4 KB	Reserved
0215_8000	0215_8FFF			4 KB	CTI
0215_7000	0215_7FFF			4 KB	Reserved
0215_6000	0215_6FFF			4 KB	Reserved
0215_5000	0215_5FFF			4 KB	Reserved
0215_4000	0215_4FFF			4 KB	Reserved
0215_3000	0215_3FFF			4 KB	Reserved
0215_2000	0215_2FFF			4 KB	Reserved
0215_1000	0215_1FFF			4 KB	PMU
0215_0000	0215_0FFF			4 KB	Debug i/f
0214_F000	0214_FFFF			4 KB	CA9-INTEG
0214_5000	0214_EFFF			40 KB	Reserved
0214_4000	0214_4FFF			4 KB	FUNNEL
0214_3000	0214_3FFF			4 KB	TPIU
0214_2000	0214_2FFF			4 KB	ext. CTI
0214_1000	0214_1FFF			4 KB	ETB
0214_0000	0214_0FFF			4 KB	DAP ROM Table

**NOTE**

User should not address reserved memory regions. Access to reserved memory regions can cause unpredictable behavior.

## 2.3 DMA memory map

The Smart DMA memory map can be found in the following table.

**Table 2-5. SDMA peripheral memory map**

Peripheral	Base address	Size
Reserved for SDMA internal memory	0x0000	4KB
SPDIF	0x1000	4KB

*Table continues on the next page...*

**Table 2-5. SDMA peripheral memory map (continued)**

Peripheral	Base address	Size
eCSPI-1	0x2000	4KB
eCSPI-2	0x3000	4KB
eCSPI-3	0x4000	4KB
eCSPI-4	0x5000	4KB
UART-5	0x6000	4KB
Reserved for SDMA internal registers	0x7000	4KB
UART-1	0x8000	4KB
UART-2	0x9000	4KB
SSI-1	0xA000	4KB
SSI-2	0xB000	4KB
SSI-3	0xC000	4KB
UART-3	0xD000	4KB
UART-4	0xE000	4KB
SPBA Registers	0xF000	4KB

**NOTE**

User should not address reserved memory regions. Access to reserved memory regions can cause unpredictable behavior.

## Chapter 3

# Interrupts and DMA Events

### 3.1 Overview

This chapter discusses the assignments of interrupts from the ARM domain in [AP interrupts](#) and from DMA events in [SDMA event mapping](#)

#### 3.1.1 AP interrupts

The Global Interrupt Controller (GIC) collects up to 128 interrupt requests from all i.MX 6SoloLite sources and provides an interface to each of the CPU cores.

The first 32 interrupts are used for interrupts that are private to the CPUs interface. These interrupts are not included in the table below. All interrupts besides the private CPU are also hooked up to the GPC in the same order.

Each interrupt can be configured as a normal or a secure interrupt. Software force registers and software priority masking are also supported. The following table describes the ARM interrupt sources.

**Table 3-1. ARM domain interrupt summary**

IRQ	Interrupt Source	Interrupt Description
32	IOMUXC	General Purpose Register 1 from IOMUXC. Used to notify core on exception condition while boot.
33	DAP	Debug Access Port interrupt request.
34	SDMA	SDMA interrupt request from all channels.
35	-	Reserved
36	SNVS	PMIC power off request.
37	RNGB	Random Number Generator interrupt request.
38	SPDC	SiPix Display Controller interrupt request.
39	CSI	CMOS Sensor Interface interrupt request

*Table continues on the next page...*

**Table 3-1. ARM domain interrupt summary (continued)**

IRQ	Interrupt Source	Interrupt Description
40	Reserved	Reserved
41	Reserved	Reserved
42	R2D GPU2D	R2D GPU2D general interrupt request.
43	V2D GPU2D	V2D GPU2D(OpenVG) general interrupt request.
44	Reserved	Reserved
45	Reserved	Reserved
46	EIM	EIM interrupt request.
47	Reserved	Reserved
48	Reserved	Reserved
49	Reserved	Reserved
50	Reserved	Reserved
51	SNVS	SNVS consolidated interrupt.
52	SNVS	SNVS security interrupt.
53	CSU	CSU interrupt request 1. Indicates to the processor that one or more alarm inputs were asserted.
54	uSDHC1	uSDHC1 (Enhanced SDHC) interrupt request.
55	uSDHC2	uSDHC2 (Enhanced SDHC) interrupt request.
56	uSDHC3	uSDHC3 (Enhanced SDHC) interrupt request.
57	uSDHC4	uSDHC4 (Enhanced SDHC) interrupt request.
58	UART1	UART1 interrupt request.
59	UART2	UART2 interrupt request.
60	UART3	UART3 interrupt request.
61	UART4	UART4 interrupt request.
62	UART5	UART5 interrupt request.
63	eCSPI1	eCSPI1 interrupt request.
64	eCSPI2	eCSPI2 interrupt request.
65	eCSPI3	eCSPI3 interrupt request.
66	eCSPI4	eCSPI4 interrupt request.
67	Reserved	Reserved
68	I2C1	I2C1 interrupt request.
69	I2C2	I2C2 interrupt request.
70	I2C3	I2C3 interrupt request.
71	LCDIF	LCDIF interrupt request.
72	USB	USB OTG2 interrupt request.
73	Reserved	Reserved
74	USB	USB Host interrupt request.
75	USB	USB OTG 1 interrupt request.
76	USB_PHY	UTMI0 interrupt request.
77	USB_PHY	UTMI1 interrupt request.

*Table continues on the next page...*



**Table 3-1. ARM domain interrupt summary (continued)**

IRQ	Interrupt Source	Interrupt Description
78	SSI1	SSI1 interrupt request.
79	SSI2	SSI2 interrupt request.
80	SSI3	SSI3 interrupt request.
81	Temperature Monitor	Temperature Sensor (temp. greater than threshold) interrupt request.
82	Reserved	Reserved
83	Reserved	Reserved
84	SPDIF	SPDIF interrupt.
85	Reserved	Reserved
86	PMU	Brown-out event on either the 1.1, 2.5 or 3.0 regulators.
87	GPT	Logical OR of GPT rollover interrupt line, input capture 1 & 2 lines, output compare 1, 2 & 3 interrupt lines.
88	EPIT1	EPIT1 output compare interrupt.
89	EPIT2	EPIT2 output compare interrupt.
90	GPIO1	INT7 interrupt request.
91	GPIO1	INT6 interrupt request.
92	GPIO1	INT5 interrupt request.
93	GPIO1	INT4 interrupt request.
94	GPIO1	INT3 interrupt request.
95	GPIO1	INT2 interrupt request.
96	GPIO1	INT1 interrupt request.
97	GPIO1	INT0 interrupt request.
98	GPIO1	Combined interrupt indication for GPIO1 signals 0 - 15.
99	GPIO1	Combined interrupt indication for GPIO1 signals 16 - 31.
100	GPIO2	Combined interrupt indication for GPIO2 signals 0 - 15.
101	GPIO2	Combined interrupt indication for GPIO2 signals 16 - 31.
102	GPIO3	Combined interrupt indication for GPIO3 signals 0 - 15.
103	GPIO3	Combined interrupt indication for GPIO3 signals 16 - 31.
104	GPIO4	Combined interrupt indication for GPIO4 signals 0 - 15.
105	GPIO4	Combined interrupt indication for GPIO4 signals 16 - 31.
106	GPIO5	Combined interrupt indication for GPIO5 signals 0 - 15.
107	GPIO5	Combined interrupt indication for GPIO5 signals 16 - 31.
108	Reserved	Reserved
109	Reserved	Reserved
110	Reserved	Reserved
111	Reserved	Reserved
112	WDOG1	WDOG1 timer reset interrupt request.
113	WDOG2	WDOG2 timer reset interrupt request.
114	KPP	Key Pad interrupt request.

*Table continues on the next page...*

**Table 3-1. ARM domain interrupt summary (continued)**

IRQ	Interrupt Source	Interrupt Description
115	PWM1	Cumulative interrupt line for PWM1. Logical OR of rollover, compare, and FIFO waterlevel crossing interrupts.
116	PWM2	Cumulative interrupt line for PWM2. Logical OR of rollover, compare, and FIFO waterlevel crossing interrupts.
117	PWM3	Cumulative interrupt line for PWM3. Logical OR of rollover, compare, and FIFO waterlevel crossing interrupts.
118	PWM4	Cumulative interrupt line for PWM4. Logical OR of rollover, compare, and FIFO waterlevel crossing interrupts.
119	CCM	CCM interrupt request 1.
120	CCM	CCM interrupt request 2.
121	GPC	GPC interrupt request 1.
122	Reserved	Reserved
123	SRC	SRC interrupt request.
124	CPU	L2 interrupt request.
125	CPU	Parity Check error interrupt request.
126	CPU	Performance Unit interrupt.
127	CPU	CTI trigger outputs interrupt.
128	SRC	CPU wdog interrupt out of SRC.
129	EPDC	EPDC interrupt request.
130	PXP	PXP interrupt request.
131	DCP	DCP general interrupt request.
132	DCP	DCP channel 0 interrupt request.
133	DCP	DCP secure interrupt request.
134	Reserved	Reserved
135	Reserved	Reserved
136	SJC	SJC interrupt from General Purpose register.
137	Reserved	Reserved
138	Reserved	Reserved
139	Reserved	Reserved
140	ASC1	ASC1 interrupt request.
141	Reserved	Reserved
142	Reserved	Reserved
143	Reserved	Reserved
144	Reserved	Reserved
145	Reserved	Reserved
146	FEC	Fast Ethernet Controller interrupt request.
147	Reserved	Reserved
148	Reserved	Reserved
149	Reserved	Reserved

*Table continues on the next page...*

**Table 3-1. ARM domain interrupt summary (continued)**

IRQ	Interrupt Source	Interrupt Description
150	Reserved	Reserved
151	Reserved	Reserved
152	Reserved	Reserved
153	Reserved	Reserved
154	Reserved	Reserved
155	Reserved	Reserved
156	Reserved	Reserved
157	Reserved	Reserved
158	Reserved	Reserved
159	PMU	Brown-out event on core, gpu or soc regulators.

## 3.2 SDMA event mapping

The following table shows the DMA request signals for peripherals in i.MX 6SoloLite.

**Table 3-2. SDMA event mapping**

Event Number	DMA Source	Description
0	Reserved	Reserved
1	Reserved	Reserved
2	Reserved	Reserved
3	eCSPI1 / I2C3	eCSPI1 Rx request; Muxed with I2C3 controlled by IOMUXC register GPR0[1].
4	eCSPI1 / I2C2	eCSPI1 Tx request; Muxed with I2C2 controlled by IOMUXC register GPR0[2].
5	eCSPI2 / I2C1	eCSPI2 Rx request; Muxed with I2C1 controlled by IOMUXC register GPR0[3].
6	eCSPI2	eCSPI2 Tx request
7	eCSPI3	eCSPI3 Rx request
8	eCSPI3	eCSPI3 Tx request
9	eCSPI4 / EPIT2	eCSPI4 Rx request; Muxed with EPIT2 DMA request controlled by IOMUXC register GPR0[5].
10	eCSPI4 / I2C1	eCSPI4 Tx request; Muxed with I2C1 controlled by IOMUXC register GPR0[4].
11	Reserved	Reserved
12	Reserved	Reserved
13	GPT	GPT counter event
14	SPDIF / IOMUX	SPDIFRX DMA request; Muxed with external DMA pad #2 controlled by IOMUXC register GPR0[7].
15	SPDIF	SPDIF TX DMA request
16	EPIT1	EPIT1 request.

*Table continues on the next page...*

**Table 3-2. SDMA event mapping (continued)**

Event Number	DMA Source	Description
17	Reserved	Reserved
18	Reserved	Reserved
19	Reserved	Reserved
20	Reserved	Reserved
21	Reserved	Reserved
22	Reserved	Reserved
23	I2C3	I2C3 DMA request
24	Reserved	Reserved
25	UART1	UART1 Rx FIFO
26	UART1	UART1 Tx FIFO
27	UART2	UART2 Rx FIFO
28	UART2	UART2 Tx FIFO
29	UART3	UART3 Rx FIFO
30	UART3	UART3 Tx FIFO
31	UART4	UART4 Rx FIFO
32	UART4	UART4 Tx FIFO
33	UART5	UART5 Rx FIFO
34	UART5	UART5 Tx FIFO
35	SSI1	SSI1 receive 1 DMA request
36	SSI1	SSI1 transmit 1 DMA request
37	SSI1	SSI1 receive 0 DMA request
38	SSI1	SSI1 transmit 0 DMA request
39	SSI2	SSI2 receive 1 DMA request
40	SSI2	SSI2 transmit 1 DMA request
41	SSI2	SSI2 receive 0 DMA request
42	SSI2	SSI2 transmit 0 DMA request
43	SSI3	SSI3 receive 1 DMA request
44	SSI3	SSI3 transmit 1 DMA request
45	SSI3	SSI3 receive 0 DMA request
46	SSI3	SSI3 transmit 0 DMA request
47	Reserved	Reserved

As shown in the table, some of the events are an output of a mux of two signals or triggers. The select of this mux is controlled by the general purpose registers in IOMUXC.

# Chapter 4

## External Signals and Pin Multiplexing

### 4.1 Overview

The i.MX 6SoloLite contains a limited number of pins, most of which have multiple signal options. These signal to pin and pin to signal options are selected by the input-output multiplexer called IOMUX. The IOMUX is also used to configure other pin characteristics, such as voltage level, drive strength, and hysteresis.

[Pin Assignments](#) lists the pad names of the chip, the various signals that can be assigned to each of the pads, and the default settings for each pad. [Muxing Options](#) lists the external signals grouped by module instance, the muxing options for each signal, and the registers used to route the signal to the chosen pad.

#### 4.1.1 Pin Assignments

Table 4-1. Pin Assignments

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
AUD_MCLK	ALT0	AUDIO_CLK_OUT	LVE - DISABLED	SW_PAD_CTL_PAD_AUD_MCLK
	ALT1	PWM4_OUT	HYS - ENABLED	
	ALT2	ECSPI3_RDY	PUS - 100K_OHM_PD	
	ALT3	FEC_MDC	PUE - KEEP	
	ALT4	WDOG2_RESET_B_DEB	PKE - ENABLED	
	ALT5	GPIO1_IO06	ODE - DISABLED	
	ALT6	SPDIF_EXT_CLK	SPEED - 100MHZ	
			DSE - 40_OHM	
			SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
AUD_RXC	ALT0	AUD3_RXC	LVE - DISABLED	SW_PAD_CTL_PAD_AUD_RXC
	ALT1	I2C1_SDA	HYS - ENABLED	
	ALT2	UART3_TX_DATA	PUS - 100K_OHM_PD	
	ALT3	FEC_TX_CLK	PUE - KEEP	
	ALT4	I2C3_SDA	PKE - ENABLED	
	ALT5	GPIO1_IO01	ODE - DISABLED	
	ALT6	ECSPI3_SS1	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
AUD_RXD	ALT0	AUD3_RXD	LVE - DISABLED	SW_PAD_CTL_PAD_AUD_RXD
	ALT1	ECSPI3_MOSI	HYS - ENABLED	
	ALT2	UART4_RX_DATA	PUS - 100K_OHM_PD	
	ALT3	FEC_RX_ER	PUE - KEEP	
	ALT4	SD1_LCTL	PKE - ENABLED	
	ALT5	GPIO1_IO02	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
AUD_RXFS	ALT0	AUD3_RXFS	LVE - DISABLED	SW_PAD_CTL_PAD_AUD_RXFS
	ALT1	I2C1_SCL	HYS - ENABLED	
	ALT2	UART3_RX_DATA	PUS - 100K_OHM_PD	
	ALT3	FEC_MDIO	PUE - KEEP	
	ALT4	I2C3_SCL	PKE - ENABLED	
	ALT5	GPIO1_IO00	ODE - DISABLED	
	ALT6	ECSPI3_SS0	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
AUD_TXC	ALT0	AUD3_TXC	LVE - DISABLED	SW_PAD_CTL_PAD_AUD_TXC
	ALT1	ECSPI3_MISO	HYS - ENABLED	
	ALT2	UART4_TX_DATA	PUS - 100K_OHM_PD	
	ALT3	FEC_RX_DV	PUE - KEEP	
	ALT4	SD2_LCTL	PKE - ENABLED	
	ALT5	GPIO1_IO03	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
AUD_TXD	ALT0	AUD3_TXD	LVE - DISABLED	SW_PAD_CTL_PAD_AUD_TXD
	ALT1	ECSPI3_SCLK	HYS - ENABLED	
	ALT2	UART4_CTS_B	PUS - 100K_OHM_PD	
	ALT3	FEC_TX_DATA0	PUE - KEEP	
	ALT4	SD4_LCTL	PKE - ENABLED	
	ALT5	GPIO1_IO05	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
AUD_TXFS	ALT0	AUD3_TXFS	LVE - DISABLED	SW_PAD_CTL_PAD_AUD_TXFS
	ALT1	PWM3_OUT	HYS - ENABLED	
	ALT2	UART4_RTS_B	PUS - 100K_OHM_PD	
	ALT3	FEC_RX_DATA1	PUE - KEEP	
	ALT4	SD3_LCTL	PKE - ENABLED	
	ALT5	GPIO1_IO04	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
BOOT_MODE0		SRC_BOOT_MODE0		
BOOT_MODE1		SRC_BOOT_MODE1		
CLK1_N		XTALOSC_CLK1_N		
CLK1_P		XTALOSC_CLK1_P		
DRAM_A0		DRAM_ADDR00	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ADDR00 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS
DRAM_A1		DRAM_ADDR01	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ADDR01 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
DRAM_A2		DRAM_ADDR02	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ADDR02 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS
DRAM_A3		DRAM_ADDR03	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ADDR03 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS
DRAM_A4		DRAM_ADDR04	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ADDR04 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS
DRAM_A5		DRAM_ADDR05	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ADDR05 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS

*Table continues on the next page...*



**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
DRAM_A6		DRAM_ADDR06	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ADDR06 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS
DRAM_A7		DRAM_ADDR07	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ADDR07 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS
DRAM_A8		DRAM_ADDR08	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ADDR08 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS
DRAM_A9		DRAM_ADDR09	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ADDR09 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS

*Table continues on the next page...*

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
DRAM_A10		DRAM_ADDR10	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ADDR10 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS
DRAM_A11		DRAM_ADDR11	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ADDR11 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS
DRAM_A12		DRAM_ADDR12	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ADDR12 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS
DRAM_A13		DRAM_ADDR13	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ADDR13 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS

*Table continues on the next page...*

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
DRAM_A14		DRAM_ADDR14	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ADDR14 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS
DRAM_A15		DRAM_ADDR15	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ADDR15 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS
DRAM_CAS		DRAM_CAS_B	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_CAS_B SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE
DRAM_CS0		DRAM_CS0_B	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_CS0_B SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_CTLDS

*Table continues on the next page...*

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
DRAM_CS1		DRAM_CS1_B	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_CS1_B SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_CTLDS
DRAM_D0		DRAM_DATA00	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B0DS
DRAM_D1		DRAM_DATA01	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B0DS
DRAM_D2		DRAM_DATA02	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B0DS
DRAM_D3		DRAM_DATA03	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B0DS

*Table continues on the next page...*

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
DRAM_D4		DRAM_DATA04	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B0DS
DRAM_D5		DRAM_DATA05	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B0DS
DRAM_D6		DRAM_DATA06	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B0DS
DRAM_D7		DRAM_DATA07	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B0DS
DRAM_D8		DRAM_DATA08	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B1DS
DRAM_D9		DRAM_DATA09	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B1DS

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
DRAM_D10		DRAM_DATA10	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B1DS
DRAM_D11		DRAM_DATA11	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B1DS
DRAM_D12		DRAM_DATA12	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B1DS
DRAM_D13		DRAM_DATA13	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B1DS
DRAM_D14		DRAM_DATA14	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B1DS
DRAM_D15		DRAM_DATA15	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B1DS

*Table continues on the next page...*

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
DRAM_D16		DRAM_DATA16	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B2DS
DRAM_D17		DRAM_DATA17	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B2DS
DRAM_D18		DRAM_DATA18	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B2DS
DRAM_D19		DRAM_DATA19	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B2DS
DRAM_D20		DRAM_DATA20	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B2DS
DRAM_D21		DRAM_DATA21	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B2DS

*Table continues on the next page...*

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
DRAM_D22		DRAM_DATA22	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B2DS
DRAM_D23		DRAM_DATA23	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B2DS
DRAM_D24		DRAM_DATA24	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B3DS
DRAM_D25		DRAM_DATA25	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B3DS
DRAM_D26		DRAM_DATA26	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B3DS
DRAM_D27		DRAM_DATA27	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B3DS

*Table continues on the next page...*



**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
DRAM_D28		DRAM_DATA28	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B3DS
DRAM_D29		DRAM_DATA29	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B3DS
DRAM_D30		DRAM_DATA30	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B3DS
DRAM_D31		DRAM_DATA31	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUE - PULL PKE - ENABLED DSE - 40_OHM	SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE SW_PAD_CTL_GRP_DDRHYS SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_B3DS
DRAM_DQM0		DRAM_DQM0	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_DQM0 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE

*Table continues on the next page...*

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
DRAM_DQM1		DRAM_DQM1	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_DQM1 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE
DRAM_DQM2		DRAM_DQM2	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_DQM2 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE
DRAM_DQM3		DRAM_DQM3	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_DQM3 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE
DRAM_SDODT0		DRAM_ODT0	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PD PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ODT0 SW_PAD_CTL_GRP_DDR_TYPE

*Table continues on the next page...*

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
DRAM_SDOdT1		DRAM_ODT1	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PD PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_ODT1 SW_PAD_CTL_GRP_DDR_TYPE
DRAM_RAS		DRAM_RAS_B	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_RAS_B SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE
DRAM_RESET		DRAM_RESET	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PD PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_RESET
DRAM_SDBA0		DRAM_SDBA0	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_SDBA0 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS

*Table continues on the next page...*

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
DRAM_SDBA1		DRAM_SDBA1	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_SDBA1 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_ADDDS
DRAM_SDBA2		DRAM_SDBA2	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_SDBA2 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_CTLDS
DRAM_SDCKE0		DRAM_SDCKE0	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PD PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_SDCKE0 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_CTLDS
DRAM_SDCKE1		DRAM_SDCKE1	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PD PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_SDCKE1 SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_CTLDS
DRAM_SDCLK_0_B		DRAM_SDCLK0_N		

*Table continues on the next page...*

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
DRAM_SDCLK_0		DRAM_SDCLK0_P	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_SDCLK0_P SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE
DRAM_SDQS0_B		DRAM_SDQS0_N		
DRAM_SDQS0		DRAM_SDQS0_P	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PD PUE - PULL PKE - DISABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_SDQS0_P SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE_CTL SW_PAD_CTL_GRP_DDRHYS
DRAM_SDQS1_B		DRAM_SDQS1_N		
DRAM_SDQS1		DRAM_SDQS1_P	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PD PUE - PULL PKE - DISABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_SDQS1_P SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE_CTL SW_PAD_CTL_GRP_DDRHYS
DRAM_SDQS2_B		DRAM_SDQS2_N		
DRAM_SDQS2		DRAM_SDQS2_P	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PD PUE - PULL PKE - DISABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_SDQS2_P SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE_CTL SW_PAD_CTL_GRP_DDRHYS
DRAM_SDQS3_B		DRAM_SDQS3_N		

*Table continues on the next page...*

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
DRAM_SDQS3		DRAM_SDQS3_P	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PD PUE - PULL PKE - DISABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_SDQS3_P SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRMODE_CTL SW_PAD_CTL_GRP_DDRHYS
DRAM_SDWE		DRAM_SDWE_B	DDR_SEL - LPDDR2 DDR_INPUT - CMOS HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	SW_PAD_CTL_PAD_DRAM_SDWE_B SW_PAD_CTL_GRP_DDR_TYPE SW_PAD_CTL_GRP_DDRPK SW_PAD_CTL_GRP_DDRPKE SW_PAD_CTL_GRP_CTLDS
ECSPI1_MISO	ALT0	ECSPI1_MISO	LVE - DISABLED	SW_PAD_CTL_PAD_ECSP11_MISO
	ALT1	AUD4_TXFS	HYS - ENABLED	
	ALT2	UART5_RTS_B	PUS - 100K_OHM_PD	
	ALT3	EPDC_BDR0	PUE - KEEP	
	ALT4	SD2_WP	PKE - ENABLED	
	ALT5	GPIO4_IO10	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
ECSPI1_MOSI	ALT0	ECSPI1_MOSI	LVE - DISABLED	SW_PAD_CTL_PAD_ECSP11_MOSI
	ALT1	AUD4_TXC	HYS - ENABLED	
	ALT2	UART5_TX_DATA	PUS - 100K_OHM_PD	
	ALT3	EPDC_VCOM1	PUE - KEEP	
	ALT4	SD2_VSELECT	PKE - ENABLED	
	ALT5	GPIO4_IO09	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
ECSPI1_SCLK	ALT0	ECSPI1_SCLK	LVE - DISABLED	SW_PAD_CTL_PAD_ECSPI1_SCLK
	ALT1	AUD4_TXD	HYS - ENABLED	
	ALT2	UART5_RX_DATA	PUS - 100K_OHM_PD	
	ALT3	EPDC_VCOM0	PUE - KEEP	
	ALT4	SD2_RESET	PKE - ENABLED	
	ALT5	GPIO4_IO08	ODE - DISABLED	
	ALT6	USB_OTG2_OC	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
ECSPI1_SS0	ALT0	ECSPI1_SS0	LVE - DISABLED	SW_PAD_CTL_PAD_ECSPI1_SS0
	ALT1	AUD4_RXD	HYS - ENABLED	
	ALT2	UART5_CTS_B	PUS - 100K_OHM_PD	
	ALT3	EPDC_BDR1	PUE - KEEP	
	ALT4	SD2_CD_B	PKE - ENABLED	
	ALT5	GPIO4_IO11	ODE - DISABLED	
	ALT6	USB_OTG2_PWR	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
ECSPI2_MISO	ALT0	ECSPI2_MISO	LVE - DISABLED	SW_PAD_CTL_PAD_ECSPI2_MISO
	ALT1	SDMA_EXT_EVENT0	HYS - ENABLED	
	ALT2	UART3_RTS_B	PUS - 100K_OHM_PD	
	ALT3	CSI_MCLK	PUE - KEEP	
	ALT4	SD1_WP	PKE - ENABLED	
	ALT5	GPIO4_IO14	ODE - DISABLED	
	ALT6	USB_OTG1_OC	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
ECSPI2_MOSI	ALT0	ECSPI2_MOSI	LVE - DISABLED	SW_PAD_CTL_PAD_ECSPI2_MOSI
	ALT1	SDMA_EXT_EVENT1	HYS - ENABLED	
	ALT2	UART3_TX_DATA	PUS - 100K_OHM_PD	
	ALT3	CSI_HSYNC	PUE - KEEP	
	ALT4	SD1_VSELECT	PKE - ENABLED	
	ALT5	GPIO4_IO13	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
ECSPI2_SCLK	ALT0	ECSPI2_SCLK	LVE - DISABLED	SW_PAD_CTL_PAD_ECSPI2_SCLK
	ALT1	SPDIF_EXT_CLK	HYS - ENABLED	
	ALT2	UART3_RX_DATA	PUS - 100K_OHM_PD	
	ALT3	CSI_PIXCLK	PUE - KEEP	
	ALT4	SD1_RESET	PKE - ENABLED	
	ALT5	GPIO4_IO12	ODE - DISABLED	
	ALT6	USB_OTG2_OC	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
ECSPI2_SS0	ALT0	ECSPI2_SS0	LVE - DISABLED	SW_PAD_CTL_PAD_ECSPI2_SS0
	ALT1	ECSPI1_SS3	HYS - ENABLED	
	ALT2	UART3_CTS_B	PUS - 100K_OHM_PD	
	ALT3	CSI_VSYNC	PUE - KEEP	
	ALT4	SD1_CD_B	PKE - ENABLED	
	ALT5	GPIO4_IO15	ODE - DISABLED	
	ALT6	USB_OTG1_PWR	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_BDR0	ALT0	EPDC_BDR0	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_BDR0
	ALT1	SD4_CLK	HYS - ENABLED	
	ALT2	UART3_RTS_B	PUS - 100K_OHM_PD	
	ALT3	EIM_ADDR26	PUE - KEEP	
	ALT4	SPDC_RL	PKE - ENABLED	
	ALT5	GPIO2_IO05	ODE - DISABLED	
	ALT6	EPDC_SDCE7	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_BDR1	ALT0	EPDC_BDR1	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_BDR1
	ALT1	SD4_CMD	HYS - ENABLED	
	ALT2	UART3_CTS_B	PUS - 100K_OHM_PD	
	ALT3	EIM_CRE	PUE - KEEP	
	ALT4	SPDC_UD	PKE - ENABLED	
	ALT5	GPIO2_IO06	ODE - DISABLED	
	ALT6	EPDC_SDCE8	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...



**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
EPDC_D0	ALT0	EPDC_DATA00	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_DATA00
	ALT1	ECSPI4_MOSI	HYS - ENABLED	
	ALT2	LCD_DATA24	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA00	PUE - KEEP	
	ALT4	SPDC_DATA00	PKE - ENABLED	
	ALT5	GPIO1_IO07	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_D1	ALT0	EPDC_DATA01	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_DATA01
	ALT1	ECSPI4_MISO	HYS - ENABLED	
	ALT2	LCD_DATA25	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA01	PUE - KEEP	
	ALT4	SPDC_DATA01	PKE - ENABLED	
	ALT5	GPIO1_IO08	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_D2	ALT0	EPDC_DATA02	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_DATA02
	ALT1	ECSPI4_SS0	HYS - ENABLED	
	ALT2	LCD_DATA26	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA02	PUE - KEEP	
	ALT4	SPDC_DATA02	PKE - ENABLED	
	ALT5	GPIO1_IO09	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_D3	ALT0	EPDC_DATA03	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_DATA03
	ALT1	ECSPI4_SCLK	HYS - ENABLED	
	ALT2	LCD_DATA27	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA03	PUE - KEEP	
	ALT4	SPDC_DATA03	PKE - ENABLED	
	ALT5	GPIO1_IO10	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

*Table continues on the next page...*

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
EPDC_D4	ALT0	EPDC_DATA04	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_DATA04
	ALT1	ECSPI4_SS1	HYS - ENABLED	
	ALT2	LCD_DATA28	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA04	PUE - KEEP	
	ALT4	SPDC_DATA04	PKE - ENABLED	
	ALT5	GPIO1_IO11	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_D5	ALT0	EPDC_DATA05	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_DATA05
	ALT1	ECSPI4_SS2	HYS - ENABLED	
	ALT2	LCD_DATA29	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA05	PUE - KEEP	
	ALT4	SPDC_DATA05	PKE - ENABLED	
	ALT5	GPIO1_IO12	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_D6	ALT0	EPDC_DATA06	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_DATA06
	ALT1	ECSPI4_SS3	HYS - ENABLED	
	ALT2	LCD_DATA30	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA06	PUE - KEEP	
	ALT4	SPDC_DATA06	PKE - ENABLED	
	ALT5	GPIO1_IO13	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_D7	ALT0	EPDC_DATA07	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_DATA07
	ALT1	ECSPI4_RDY	HYS - ENABLED	
	ALT2	LCD_DATA31	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA07	PUE - KEEP	
	ALT4	SPDC_DATA07	PKE - ENABLED	
	ALT5	GPIO1_IO14	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
EPDC_D8	ALT0	EPDC_DATA08	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_DATA08
	ALT1	ECSPI3_MOSI	HYS - ENABLED	
	ALT2	EPDC_PWR_CTRL0	PUS - 100K_OHM_PD	
	ALT3	EIM_ADDR16	PUE - KEEP	
	ALT4	SPDC_DATA08	PKE - ENABLED	
	ALT5	GPIO1_IO15	ODE - DISABLED	
	ALT6	SD4_RESET	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_D9	ALT0	EPDC_DATA09	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_DATA09
	ALT1	ECSPI3_MISO	HYS - ENABLED	
	ALT2	EPDC_PWR_CTRL1	PUS - 100K_OHM_PD	
	ALT3	EIM_ADDR17	PUE - KEEP	
	ALT4	SPDC_DATA09	PKE - ENABLED	
	ALT5	GPIO1_IO16	ODE - DISABLED	
	ALT6	SD4_VSELECT	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_D10	ALT0	EPDC_DATA10	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_DATA10
	ALT1	ECSPI3_SS0	HYS - ENABLED	
	ALT2	EPDC_PWR_CTRL2	PUS - 100K_OHM_PD	
	ALT3	EIM_ADDR18	PUE - KEEP	
	ALT4	SPDC_DATA10	PKE - ENABLED	
	ALT5	GPIO1_IO17	ODE - DISABLED	
	ALT6	SD4_WP	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_D11	ALT0	EPDC_DATA11	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_DATA11
	ALT1	ECSPI3_SCLK	HYS - ENABLED	
	ALT2	EPDC_PWR_CTRL3	PUS - 100K_OHM_PD	
	ALT3	EIM_ADDR19	PUE - KEEP	
	ALT4	SPDC_DATA11	PKE - ENABLED	
	ALT5	GPIO1_IO18	ODE - DISABLED	
	ALT6	SD4_CD_B	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
EPDC_D12	ALT0	EPDC_DATA12	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_DATA12
	ALT1	UART2_RX_DATA	HYS - ENABLED	
	ALT2	EPDC_PWR_COM	PUS - 100K_OHM_PD	
	ALT3	EIM_ADDR20	PUE - KEEP	
	ALT4	SPDC_DATA12	PKE - ENABLED	
	ALT5	GPIO1_IO19	ODE - DISABLED	
	ALT6	ECSPI3_SS1	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_D13	ALT0	EPDC_DATA13	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_DATA13
	ALT1	UART2_TX_DATA	HYS - ENABLED	
	ALT2	EPDC_PWR_IRQ	PUS - 100K_OHM_PD	
	ALT3	EIM_ADDR21	PUE - KEEP	
	ALT4	SPDC_DATA13	PKE - ENABLED	
	ALT5	GPIO1_IO20	ODE - DISABLED	
	ALT6	ECSPI3_SS2	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_D14	ALT0	EPDC_DATA14	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_DATA14
	ALT1	UART2_RTS_B	HYS - ENABLED	
	ALT2	EPDC_PWR_STAT	PUS - 100K_OHM_PD	
	ALT3	EIM_ADDR22	PUE - KEEP	
	ALT4	SPDC_DATA14	PKE - ENABLED	
	ALT5	GPIO1_IO21	ODE - DISABLED	
	ALT6	ECSPI3_SS3	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_D15	ALT0	EPDC_DATA15	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_DATA15
	ALT1	UART2_CTS_B	HYS - ENABLED	
	ALT2	EPDC_PWR_WAKE	PUS - 100K_OHM_PD	
	ALT3	EIM_ADDR23	PUE - KEEP	
	ALT4	SPDC_DATA15	PKE - ENABLED	
	ALT5	GPIO1_IO22	ODE - DISABLED	
	ALT6	ECSPI3_RDY	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
EPDC_GDCLK	ALT0	EPDC_GDCLK	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_GDCLK
	ALT1	ECSPI2_SS2	HYS - ENABLED	
	ALT2	SPDC_YCKR	PUS - 100K_OHM_PD	
	ALT3	CSI_PIXCLK	PUE - KEEP	
	ALT4	SPDC_YCKL	PKE - ENABLED	
	ALT5	GPIO1_IO31	ODE - DISABLED	
	ALT6	SD2_RESET	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_GDOE	ALT0	EPDC_GDOE	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_GDOE
	ALT1	ECSPI2_SS3	HYS - ENABLED	
	ALT2	SPDC_YOER	PUS - 100K_OHM_PD	
	ALT3	CSI_HSYNC	PUE - KEEP	
	ALT4	SPDC_YOEL	PKE - ENABLED	
	ALT5	GPIO2_IO00	ODE - DISABLED	
	ALT6	SD2_VSELECT	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_GDRL	ALT0	EPDC_GDRL	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_GDRL
	ALT1	ECSPI2_RDY	HYS - ENABLED	
	ALT2	SPDC_YDIOUR	PUS - 100K_OHM_PD	
	ALT3	CSI_MCLK	PUE - KEEP	
	ALT4	SPDC_YDIOUL	PKE - ENABLED	
	ALT5	GPIO2_IO01	ODE - DISABLED	
	ALT6	SD2_WP	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_GDSP	ALT0	EPDC_GDSP	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_GDSP
	ALT1	PWM4_OUT	HYS - ENABLED	
	ALT2	SPDC_YDIOUR	PUS - 100K_OHM_PD	
	ALT3	CSI_VSYNC	PUE - KEEP	
	ALT4	SPDC_YDIOUL	PKE - ENABLED	
	ALT5	GPIO2_IO02	ODE - DISABLED	
	ALT6	SD2_CD_B	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
EPDC_PWRCOM	ALT0	EPDC_PWR_COM	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_PWR_COM
	ALT1	SD4_DATA0	HYS - ENABLED	
	ALT2	LCD_DATA20	PUS - 100K_OHM_PD	
	ALT3	EIM_BCLK	PUE - KEEP	
	ALT4	USB_OTG1_ID	PKE - ENABLED	
	ALT5	GPIO2_IO11	ODE - DISABLED	
	ALT6	SD3_RESET	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_PWRCTRL 0	ALT0	EPDC_PWR_CTRL0	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_PWR_CTRL 0
	ALT1	AUD5_RXC	HYS - ENABLED	
	ALT2	LCD_DATA16	PUS - 100K_OHM_PD	
	ALT3	EIM_RW	PUE - KEEP	
	ALT4	SPDC_YCKL	PKE - ENABLED	
	ALT5	GPIO2_IO07	ODE - DISABLED	
	ALT6	SD4_RESET	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_PWRCTRL 1	ALT0	EPDC_PWR_CTRL1	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_PWR_CTRL 1
	ALT1	AUD5_TXFS	HYS - ENABLED	
	ALT2	LCD_DATA17	PUS - 100K_OHM_PD	
	ALT3	EIM_OE_B	PUE - KEEP	
	ALT4	SPDC_YOEL	PKE - ENABLED	
	ALT5	GPIO2_IO08	ODE - DISABLED	
	ALT6	SD4_VSELECT	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_PWRCTRL 2	ALT0	EPDC_PWR_CTRL2	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_PWR_CTRL 2
	ALT1	AUD5_TXD	HYS - ENABLED	
	ALT2	LCD_DATA18	PUS - 100K_OHM_PD	
	ALT3	EIM_CS0_B	PUE - KEEP	
	ALT4	SPDC_YDIOUL	PKE - ENABLED	
	ALT5	GPIO2_IO09	ODE - DISABLED	
	ALT6	SD4_WP	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
EPDC_PWRCTRL 3	ALT0	EPDC_PWR_CTRL3	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_PWR_CTRL 3
	ALT1	AUD5_TXC	HYS - ENABLED	
	ALT2	LCD_DATA19	PUS - 100K_OHM_PD	
	ALT3	EIM_CS1_B	PUE - KEEP	
	ALT4	SPDC_YDIODL	PKE - ENABLED	
	ALT5	GPIO2_IO10	ODE - DISABLED	
	ALT6	SD4_CD_B	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_PWRINT	ALT0	EPDC_PWR_IRQ	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_PWR_IRQ
	ALT1	SD4_DATA1	HYS - ENABLED	
	ALT2	LCD_DATA21	PUS - 100K_OHM_PD	
	ALT3	EIM_ACLK_FREERUN	PUE - KEEP	
	ALT4	USB_OTG2_ID	PKE - ENABLED	
	ALT5	GPIO2_IO12	ODE - DISABLED	
	ALT6	SD3_VSELECT	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_PWRSTAT	ALT0	EPDC_PWR_STAT	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_PWR_STAT
	ALT1	SD4_DATA2	HYS - ENABLED	
	ALT2	LCD_DATA22	PUS - 100K_OHM_PD	
	ALT3	EIM_WAIT_B	PUE - KEEP	
	ALT4	ARM_EVENTI	PKE - ENABLED	
	ALT5	GPIO2_IO13	ODE - DISABLED	
	ALT6	SD3_WP	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_PWRWAK EUP	ALT0	EPDC_PWR_WAKE	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_PWR_WAK E
	ALT1	SD4_DATA3	HYS - ENABLED	
	ALT2	LCD_DATA23	PUS - 100K_OHM_PD	
	ALT3	EIM_DTACK_B	PUE - KEEP	
	ALT4	ARM_EVENTO	PKE - ENABLED	
	ALT5	GPIO2_IO14	ODE - DISABLED	
	ALT6	SD3_CD_B	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
EPDC_SDCE0	ALT0	EPDC_SDCE0	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_SDCE0
	ALT1	ECSPi2_SS1	HYS - ENABLED	
	ALT2	PWM3_OUT	PUS - 100K_OHM_PD	
	ALT3	EIM_CS2_B	PUE - KEEP	
	ALT4	SPDC_YCKR	PKE - ENABLED	
	ALT5	GPIO1_IO27	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_SDCE1	ALT0	EPDC_SDCE1	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_SDCE1
	ALT1	WD0G2_B	HYS - ENABLED	
	ALT2	PWM4_OUT	PUS - 100K_OHM_PD	
	ALT3	EIM_LBA_B	PUE - KEEP	
	ALT4	SPDC_YOER	PKE - ENABLED	
	ALT5	GPIO1_IO28	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_SDCE2	ALT0	EPDC_SDCE2	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_SDCE2
	ALT1	I2C3_SCL	HYS - ENABLED	
	ALT2	PWM1_OUT	PUS - 100K_OHM_PD	
	ALT3	EIM_EB0_B	PUE - KEEP	
	ALT4	SPDC_YDIOUR	PKE - ENABLED	
	ALT5	GPIO1_IO29	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_SDCE3	ALT0	EPDC_SDCE3	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_SDCE3
	ALT1	I2C3_SDA	HYS - ENABLED	
	ALT2	PWM2_OUT	PUS - 100K_OHM_PD	
	ALT3	EIM_EB1_B	PUE - KEEP	
	ALT4	SPDC_YDIOUR	PKE - ENABLED	
	ALT5	GPIO1_IO30	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...



**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
EPDC_SDCLK	ALT0	EPDC_SDCLK_P	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_SDCLK
	ALT1	ECSPI2_MOSI	HYS - ENABLED	
	ALT2	I2C2_SCL	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA08	PUE - KEEP	
	ALT4	SPDC_CL	PKE - ENABLED	
	ALT5	GPIO1_IO23	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_SDLE	ALT0	EPDC_SDLE	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_SDLE
	ALT1	ECSPI2_MISO	HYS - ENABLED	
	ALT2	I2C2_SDA	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA09	PUE - KEEP	
	ALT4	SPDC_LD	PKE - ENABLED	
	ALT5	GPIO1_IO24	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_SDOE	ALT0	EPDC_SDOE	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_SDOE
	ALT1	ECSPI2_SS0	HYS - ENABLED	
	ALT2	SPDC_XDIOR	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA10	PUE - KEEP	
	ALT4	SPDC_XDIOL	PKE - ENABLED	
	ALT5	GPIO1_IO25	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_SDSHR	ALT0	EPDC_SDSHR	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_SDSHR
	ALT1	ECSPI2_SCLK	HYS - ENABLED	
	ALT2	EPDC_SDCE4	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA11	PUE - KEEP	
	ALT4	SPDC_XDIOR	PKE - ENABLED	
	ALT5	GPIO1_IO26	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
EPDC_VCOM0	ALT0	EPDC_VCOM0	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_VCOM0
	ALT1	AUD5_RXFS	HYS - ENABLED	
	ALT2	UART3_RX_DATA	PUS - 100K_OHM_PD	
	ALT3	EIM_ADDR24	PUE - KEEP	
	ALT4	SPDC_VCOM0	PKE - ENABLED	
	ALT5	GPIO2_IO03	ODE - DISABLED	
	ALT6	EPDC_SDCE5	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
EPDC_VCOM1	ALT0	EPDC_VCOM1	LVE - DISABLED	SW_PAD_CTL_PAD_EPDC_VCOM1
	ALT1	AUD5_RXD	HYS - ENABLED	
	ALT2	UART3_TX_DATA	PUS - 100K_OHM_PD	
	ALT3	EIM_ADDR25	PUE - KEEP	
	ALT4	SPDC_VCOM1	PKE - ENABLED	
	ALT5	GPIO2_IO04	ODE - DISABLED	
	ALT6	EPDC_SDCE6	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
FEC_CRS_DV	ALT0	FEC_RX_DV	LVE - DISABLED	SW_PAD_CTL_PAD_FEC_CRS_DV
	ALT1	SD4_DATA1	HYS - ENABLED	
	ALT2	AUD6_TXC	PUS - 100K_OHM_PD	
	ALT3	ECSPi4_MISO	PUE - KEEP	
	ALT4	GPT_COMPARE2	PKE - ENABLED	
	ALT5	GPIO4_IO25	ODE - DISABLED	
	ALT6	ARM_TRACE31	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
FEC_MDC	ALT0	FEC_MDC	LVE - DISABLED	SW_PAD_CTL_PAD_FEC_MDC
	ALT1	SD4_DATA4	HYS - ENABLED	
	ALT2	AUDIO_CLK_OUT	PUS - 100K_OHM_PD	
	ALT3	SD1_RESET	PUE - KEEP	
	ALT4	SD3_RESET	PKE - ENABLED	
	ALT5	GPIO4_IO23	ODE - DISABLED	
	ALT6	ARM_TRACE29	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
FEC_MDIO	ALT0	FEC_MDIO	LVE - DISABLED	SW_PAD_CTL_PAD_FEC_MDIO
	ALT1	SD4_CLK	HYS - ENABLED	
	ALT2	AUD6_RXFS	PUS - 100K_OHM_PD	
	ALT3	ECSPI4_SS0	PUE - KEEP	
	ALT4	GPT_CAPTURE1	PKE - ENABLED	
	ALT5	GPIO4_IO20	ODE - DISABLED	
	ALT6	ARM_TRACE26	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
FEC_REF_CLK	ALT0	FEC_REF_OUT	LVE - DISABLED	SW_PAD_CTL_PAD_FEC_REF_CLK
	ALT1	SD4_RESET	HYS - ENABLED	
	ALT2	WDOG1_B	PUS - 100K_OHM_PD	
	ALT3	PWM4_OUT	PUE - KEEP	
	ALT4	CCM_PMIC_READY	PKE - ENABLED	
	ALT5	GPIO4_IO26	ODE - DISABLED	
	ALT6	SPDIF_EXT_CLK	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
FEC_RXD0	ALT0	FEC_RX_DATA0	LVE - DISABLED	SW_PAD_CTL_PAD_FEC_RX_DATA0
	ALT1	SD4_DATA5	HYS - ENABLED	
	ALT2	USB_OTG1_ID	PUS - 100K_OHM_PD	
	ALT3	SD1_VSELECT	PUE - KEEP	
	ALT4	SD3_VSELECT	PKE - ENABLED	
	ALT5	GPIO4_IO17	ODE - DISABLED	
	ALT6	ARM_TRACE24	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
FEC_RXD1	ALT0	FEC_RX_DATA1	LVE - DISABLED	SW_PAD_CTL_PAD_FEC_RX_DATA1
	ALT1	SD4_DATA2	HYS - ENABLED	
	ALT2	AUD6_TXFS	PUS - 100K_OHM_PD	
	ALT3	ECSPI4_SS1	PUE - KEEP	
	ALT4	GPT_COMPARE3	PKE - ENABLED	
	ALT5	GPIO4_IO18	ODE - DISABLED	
	ALT6	FEC_COL	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
FEC_RX_ER	ALT0	FEC_RX_ER	LVE - DISABLED	SW_PAD_CTL_PAD_FEC_RX_ER
	ALT1	SD4_DATA0	HYS - ENABLED	
	ALT2	AUD6_RXD	PUS - 100K_OHM_PD	
	ALT3	ECSPI4_MOSI	PUE - KEEP	
	ALT4	GPT_COMPARE1	PKE - ENABLED	
	ALT5	GPIO4_IO19	ODE - DISABLED	
	ALT6	ARM_TRACE25	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
FEC_TX_CLK	ALT0	FEC_TX_CLK	LVE - DISABLED	SW_PAD_CTL_PAD_FEC_TX_CLK
	ALT1	SD4_CMD	HYS - ENABLED	
	ALT2	AUD6_RXC	PUS - 100K_OHM_PD	
	ALT3	ECSPI4_SCLK	PUE - KEEP	
	ALT4	GPT_CAPTURE2	PKE - ENABLED	
	ALT5	GPIO4_IO21	ODE - DISABLED	
	ALT6	ARM_TRACE27	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
FEC_TXD0	ALT0	FEC_TX_DATA0	LVE - DISABLED	SW_PAD_CTL_PAD_FEC_TX_DATA0
	ALT1	SD4_DATA3	HYS - ENABLED	
	ALT2	AUD6_TXD	PUS - 100K_OHM_PD	
	ALT3	ECSPI4_SS2	PUE - KEEP	
	ALT4	GPT_CLKIN	PKE - ENABLED	
	ALT5	GPIO4_IO24	ODE - DISABLED	
	ALT6	ARM_TRACE30	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
FEC_TXD1	ALT0	FEC_TX_DATA1	LVE - DISABLED	SW_PAD_CTL_PAD_FEC_TX_DATA1
	ALT1	SD4_DATA7	HYS - ENABLED	
	ALT2	SPDIF_OUT	PUS - 100K_OHM_PD	
	ALT3	SD1_CD_B	PUE - KEEP	
	ALT4	SD3_CD_B	PKE - ENABLED	
	ALT5	GPIO4_IO16	ODE - DISABLED	
	ALT6	FEC_RX_CLK	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
FEC_TX_EN	ALT0	FEC_TX_EN	LVE - DISABLED	SW_PAD_CTL_PAD_FEC_TX_EN
	ALT1	SD4_DATA6	HYS - ENABLED	
	ALT2	SPDIF_IN	PUS - 100K_OHM_PD	
	ALT3	SD1_WP	PUE - KEEP	
	ALT4	SD3_WP	PKE - ENABLED	
	ALT5	GPIO4_IO22	ODE - DISABLED	
	ALT6	ARM_TRACE28	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
I2C1_SCL	ALT0	I2C1_SCL	LVE - DISABLED	SW_PAD_CTL_PAD_I2C1_SCL
	ALT1	UART1_RTS_B	HYS - ENABLED	
	ALT2	ECSPI3_SS2	PUS - 100K_OHM_PD	
	ALT3	FEC_RX_DATA0	PUE - KEEP	
	ALT4	SD3_RESET	PKE - ENABLED	
	ALT5	GPIO3_IO12	ODE - DISABLED	
	ALT6	ECSPI1_SS1	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
I2C1_SDA	ALT0	I2C1_SDA	LVE - DISABLED	SW_PAD_CTL_PAD_I2C1_SDA
	ALT1	UART1_CTS_B	HYS - ENABLED	
	ALT2	ECSPI3_SS3	PUS - 100K_OHM_PD	
	ALT3	FEC_TX_EN	PUE - KEEP	
	ALT4	SD3_VSELECT	PKE - ENABLED	
	ALT5	GPIO3_IO13	ODE - DISABLED	
	ALT6	ECSPI1_SS2	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
I2C2_SCL	ALT0	I2C2_SCL	LVE - DISABLED	SW_PAD_CTL_PAD_I2C2_SCL
	ALT1	AUD4_RXFS	HYS - ENABLED	
	ALT2	SPDIF_IN	PUS - 100K_OHM_PD	
	ALT3	FEC_TX_DATA1	PUE - KEEP	
	ALT4	SD3_WP	PKE - ENABLED	
	ALT5	GPIO3_IO14	ODE - DISABLED	
	ALT6	ECSPI1_RDY	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
I2C2_SDA	ALT0	I2C2_SDA	LVE - DISABLED	SW_PAD_CTL_PAD_I2C2_SDA
	ALT1	AUD4_RXC	HYS - ENABLED	
	ALT2	SPDIF_OUT	PUS - 100K_OHM_PD	
	ALT3	FEC_REF_OUT	PUE - KEEP	
	ALT4	SD3_CD_B	PKE - ENABLED	
	ALT5	GPIO3_IO15	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
JTAG_MOD		JTAG_MOD	LVE - DISABLED HYS - DISABLED PUS - 100K_OHM_PU PUE - PULL PKE - ENABLED ODE - DISABLED SPEED - 50MHZ DSE - 60_OHM SRE - SLOW	SW_PAD_CTL_PAD_JTAG_MOD
JTAG_TCK		JTAG_TCK	LVE - DISABLED HYS - DISABLED PUS - 47K_OHM_PU PUE - PULL PKE - ENABLED ODE - DISABLED SPEED - 50MHZ DSE - 60_OHM SRE - SLOW	SW_PAD_CTL_PAD_JTAG_TCK
JTAG_TDI		JTAG_TDI	LVE - DISABLED HYS - DISABLED PUS - 47K_OHM_PU PUE - PULL PKE - ENABLED ODE - DISABLED SPEED - 50MHZ DSE - 60_OHM SRE - SLOW	SW_PAD_CTL_PAD_JTAG_TDI

*Table continues on the next page...*

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
JTAG_TDO		JTAG_TDO	LVE - DISABLED HYS - DISABLED PUS - 100K_OHM_PU PUE - KEEP PKE - ENABLED ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - FAST	SW_PAD_CTL_PAD_JTAG_TDO
JTAG_TMS		JTAG_TMS	LVE - DISABLED HYS - DISABLED PUS - 47K_OHM_PU PUE - PULL PKE - ENABLED ODE - DISABLED SPEED - 50MHZ DSE - 60_OHM SRE - SLOW	SW_PAD_CTL_PAD_JTAG_TMS
JTAG_TRSTB		JTAG_TRSTB	LVE - DISABLED HYS - DISABLED PUS - 47K_OHM_PU PUE - PULL PKE - ENABLED ODE - DISABLED SPEED - 50MHZ DSE - 60_OHM SRE - SLOW	SW_PAD_CTL_PAD_JTAG_TRSTB
KEY_COL0	ALT0	KEY_COL0	LVE - DISABLED	SW_PAD_CTL_PAD_KEY_COL0
	ALT1	I2C2_SCL	HYS - ENABLED	
	ALT2	LCD_DATA00	PUS - 100K_OHM_PD	
	ALT3	EIM_AD00	PUE - KEEP	
	ALT4	SD1_CD_B	PKE - ENABLED	
	ALT5	GPIO3_IO24	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
KEY_COL1	ALT0	KEY_COL1	LVE - DISABLED	SW_PAD_CTL_PAD_KEY_COL1
	ALT1	ECSPI4_MOSI	HYS - ENABLED	
	ALT2	LCD_DATA02	PUS - 100K_OHM_PD	
	ALT3	EIM_AD02	PUE - KEEP	
	ALT4	SD3_DATA4	PKE - ENABLED	
	ALT5	GPIO3_IO26	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
KEY_COL2	ALT0	KEY_COL2	LVE - DISABLED	SW_PAD_CTL_PAD_KEY_COL2
	ALT1	ECSPI4_SS0	HYS - ENABLED	
	ALT2	LCD_DATA04	PUS - 100K_OHM_PD	
	ALT3	EIM_AD04	PUE - KEEP	
	ALT4	SD3_DATA6	PKE - ENABLED	
	ALT5	GPIO3_IO28	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
KEY_COL3	ALT0	KEY_COL3	LVE - DISABLED	SW_PAD_CTL_PAD_KEY_COL3
	ALT1	AUD6_RXFS	HYS - ENABLED	
	ALT2	LCD_DATA06	PUS - 100K_OHM_PD	
	ALT3	EIM_AD06	PUE - KEEP	
	ALT4	SD4_DATA6	PKE - ENABLED	
	ALT5	GPIO3_IO30	ODE - DISABLED	
	ALT6	SD1_RESET	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
KEY_COL4	ALT0	KEY_COL4	LVE - DISABLED	SW_PAD_CTL_PAD_KEY_COL4
	ALT1	AUD6_RXD	HYS - ENABLED	
	ALT2	LCD_DATA08	PUS - 100K_OHM_PD	
	ALT3	EIM_AD08	PUE - KEEP	
	ALT4	SD4_CLK	PKE - ENABLED	
	ALT5	GPIO4_IO00	ODE - DISABLED	
	ALT6	USB_OTG1_PWR	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...



**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
KEY_COL5	ALT0	KEY_COL5	LVE - DISABLED	SW_PAD_CTL_PAD_KEY_COL5
	ALT1	AUD6_TXFS	HYS - ENABLED	
	ALT2	LCD_DATA10	PUS - 100K_OHM_PD	
	ALT3	EIM_AD10	PUE - KEEP	
	ALT4	SD4_DATA0	PKE - ENABLED	
	ALT5	GPIO4_IO02	ODE - DISABLED	
	ALT6	USB_OTG2_PWR	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
KEY_COL6	ALT0	KEY_COL6	LVE - DISABLED	SW_PAD_CTL_PAD_KEY_COL6
	ALT1	UART4_RX_DATA	HYS - ENABLED	
	ALT2	LCD_DATA12	PUS - 100K_OHM_PD	
	ALT3	EIM_AD12	PUE - KEEP	
	ALT4	SD4_DATA2	PKE - ENABLED	
	ALT5	GPIO4_IO04	ODE - DISABLED	
	ALT6	SD3_RESET	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
KEY_COL7	ALT0	KEY_COL7	LVE - DISABLED	SW_PAD_CTL_PAD_KEY_COL7
	ALT1	UART4_RTS_B	HYS - ENABLED	
	ALT2	LCD_DATA14	PUS - 100K_OHM_PD	
	ALT3	EIM_AD14	PUE - KEEP	
	ALT4	SD4_DATA4	PKE - ENABLED	
	ALT5	GPIO4_IO06	ODE - DISABLED	
	ALT6	SD1_WP	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
KEY_ROW0	ALT0	KEY_ROW0	LVE - DISABLED	SW_PAD_CTL_PAD_KEY_ROW0
	ALT1	I2C2_SDA	HYS - ENABLED	
	ALT2	LCD_DATA01	PUS - 100K_OHM_PD	
	ALT3	EIM_AD01	PUE - KEEP	
	ALT4	SD1_WP	PKE - ENABLED	
	ALT5	GPIO3_IO25	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
KEY_ROW1	ALT0	KEY_ROW1	LVE - DISABLED	SW_PAD_CTL_PAD_KEY_ROW1
	ALT1	ECSPi4_MISO	HYS - ENABLED	
	ALT2	LCD_DATA03	PUS - 100K_OHM_PD	
	ALT3	EIM_AD03	PUE - KEEP	
	ALT4	SD3_DATA5	PKE - ENABLED	
	ALT5	GPIO3_IO27	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
KEY_ROW2	ALT0	KEY_ROW2	LVE - DISABLED	SW_PAD_CTL_PAD_KEY_ROW2
	ALT1	ECSPi4_SCLK	HYS - ENABLED	
	ALT2	LCD_DATA05	PUS - 100K_OHM_PD	
	ALT3	EIM_AD05	PUE - KEEP	
	ALT4	SD3_DATA7	PKE - ENABLED	
	ALT5	GPIO3_IO29	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
KEY_ROW3	ALT0	KEY_ROW3	LVE - DISABLED	SW_PAD_CTL_PAD_KEY_ROW3
	ALT1	AUD6_RXC	HYS - ENABLED	
	ALT2	LCD_DATA07	PUS - 100K_OHM_PD	
	ALT3	EIM_AD07	PUE - KEEP	
	ALT4	SD4_DATA7	PKE - ENABLED	
	ALT5	GPIO3_IO31	ODE - DISABLED	
	ALT6	SD1_VSELECT	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
KEY_ROW4	ALT0	KEY_ROW4	LVE - DISABLED	SW_PAD_CTL_PAD_KEY_ROW4
	ALT1	AUD6_TXC	HYS - ENABLED	
	ALT2	LCD_DATA09	PUS - 100K_OHM_PD	
	ALT3	EIM_AD09	PUE - KEEP	
	ALT4	SD4_CMD	PKE - ENABLED	
	ALT5	GPIO4_IO01	ODE - DISABLED	
	ALT6	USB_OTG1_OC	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
KEY_ROW5	ALT0	KEY_ROW5	LVE - DISABLED	SW_PAD_CTL_PAD_KEY_ROW5
	ALT1	AUD6_TXD	HYS - ENABLED	
	ALT2	LCD_DATA11	PUS - 100K_OHM_PD	
	ALT3	EIM_AD11	PUE - KEEP	
	ALT4	SD4_DATA1	PKE - ENABLED	
	ALT5	GPIO4_IO03	ODE - DISABLED	
	ALT6	USB_OTG2_OC	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
KEY_ROW6	ALT0	KEY_ROW6	LVE - DISABLED	SW_PAD_CTL_PAD_KEY_ROW6
	ALT1	UART4_TX_DATA	HYS - ENABLED	
	ALT2	LCD_DATA13	PUS - 100K_OHM_PD	
	ALT3	EIM_AD13	PUE - KEEP	
	ALT4	SD4_DATA3	PKE - ENABLED	
	ALT5	GPIO4_IO05	ODE - DISABLED	
	ALT6	SD3_VSELECT	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
KEY_ROW7	ALT0	KEY_ROW7	LVE - DISABLED	SW_PAD_CTL_PAD_KEY_ROW7
	ALT1	UART4_CTS_B	HYS - ENABLED	
	ALT2	LCD_DATA15	PUS - 100K_OHM_PD	
	ALT3	EIM_AD15	PUE - KEEP	
	ALT4	SD4_DATA5	PKE - ENABLED	
	ALT5	GPIO4_IO07	ODE - DISABLED	
	ALT6	SD1_CD_B	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
LCD_CLK	ALT0	LCD_CLK	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_CLK
	ALT1	SD4_DATA4	HYS - ENABLED	
	ALT2	LCD_WR_RWN	PUS - 100K_OHM_PD	
	ALT3	EIM_RW	PUE - KEEP	
	ALT4	PWM4_OUT	PKE - ENABLED	
	ALT5	GPIO2_IO15	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
LCD_DAT0	ALT0	LCD_DATA00	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA00
	ALT1	ECSPI1_MOSI	HYS - ENABLED	
	ALT2	USB_OTG2_ID	PUS - 100K_OHM_PD	
	ALT3	PWM1_OUT	PUE - KEEP	
	ALT4	UART5_DTR_B	PKE - ENABLED	
	ALT5	GPIO2_IO20	ODE - DISABLED	
	ALT6	ARM_TRACE00	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG00	DSE - 40_OHM SRE - SLOW	
LCD_DAT1	ALT0	LCD_DATA01	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA01
	ALT1	ECSPI1_MISO	HYS - ENABLED	
	ALT2	USB_OTG1_ID	PUS - 100K_OHM_PD	
	ALT3	PWM2_OUT	PUE - KEEP	
	ALT4	AUD4_RXFS	PKE - ENABLED	
	ALT5	GPIO2_IO21	ODE - DISABLED	
	ALT6	ARM_TRACE01	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG01	DSE - 40_OHM SRE - SLOW	
LCD_DAT2	ALT0	LCD_DATA02	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA02
	ALT1	ECSPI1_SS0	HYS - ENABLED	
	ALT2	EPIT2_OUT	PUS - 100K_OHM_PD	
	ALT3	PWM3_OUT	PUE - KEEP	
	ALT4	AUD4_RXC	PKE - ENABLED	
	ALT5	GPIO2_IO22	ODE - DISABLED	
	ALT6	ARM_TRACE02	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG02	DSE - 40_OHM SRE - SLOW	
LCD_DAT3	ALT0	LCD_DATA03	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA03
	ALT1	ECSPI1_SCLK	HYS - ENABLED	
	ALT2	UART5_DSR_B	PUS - 100K_OHM_PD	
	ALT3	PWM4_OUT	PUE - KEEP	
	ALT4	AUD4_RXD	PKE - ENABLED	
	ALT5	GPIO2_IO23	ODE - DISABLED	
	ALT6	ARM_TRACE03	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG03	DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
LCD_DAT4	ALT0	LCD_DATA04	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA04
	ALT1	ECSPI1_SS1	HYS - ENABLED	
	ALT2	CSI_VSYNC	PUS - 100K_OHM_PD	
	ALT3	WDOG2_RESET_B_DEB	PUE - KEEP	
	ALT4	AUD4_TXC	PKE - ENABLED	
	ALT5	GPIO2_IO24	ODE - DISABLED	
	ALT6	ARM_TRACE04	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG04	DSE - 40_OHM SRE - SLOW	
LCD_DAT5	ALT0	LCD_DATA05	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA05
	ALT1	ECSPI1_SS2	HYS - ENABLED	
	ALT2	CSI_HSYNC	PUS - 100K_OHM_PD	
	ALT3	EIM_CS3_B	PUE - KEEP	
	ALT4	AUD4_TXFS	PKE - ENABLED	
	ALT5	GPIO2_IO25	ODE - DISABLED	
	ALT6	ARM_TRACE05	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG05	DSE - 40_OHM SRE - SLOW	
LCD_DAT6	ALT0	LCD_DATA06	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA06
	ALT1	ECSPI1_SS3	HYS - ENABLED	
	ALT2	CSI_PIXCLK	PUS - 100K_OHM_PD	
	ALT3	EIM_DATA00	PUE - KEEP	
	ALT4	AUD4_TXD	PKE - ENABLED	
	ALT5	GPIO2_IO26	ODE - DISABLED	
	ALT6	ARM_TRACE06	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG06	DSE - 40_OHM SRE - SLOW	
LCD_DAT7	ALT0	LCD_DATA07	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA07
	ALT1	ECSPI1_RDY	HYS - ENABLED	
	ALT2	CSI_MCLK	PUS - 100K_OHM_PD	
	ALT3	EIM_DATA01	PUE - KEEP	
	ALT4	AUDIO_CLK_OUT	PKE - ENABLED	
	ALT5	GPIO2_IO27	ODE - DISABLED	
	ALT6	ARM_TRACE07	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG07	DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
LCD_DAT8	ALT0	LCD_DATA08	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA08
	ALT1	KEY_COL0	HYS - ENABLED	
	ALT2	CSI_DATA09	PUS - 100K_OHM_PD	
	ALT3	EIM_DATA02	PUE - KEEP	
	ALT4	ECSPi2_SCLK	PKE - ENABLED	
	ALT5	GPIO2_IO28	ODE - DISABLED	
	ALT6	ARM_TRACE08	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG08	DSE - 40_OHM SRE - SLOW	
LCD_DAT9	ALT0	LCD_DATA09	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA09
	ALT1	KEY_ROW0	HYS - ENABLED	
	ALT2	CSI_DATA08	PUS - 100K_OHM_PD	
	ALT3	EIM_DATA03	PUE - KEEP	
	ALT4	ECSPi2_MOSI	PKE - ENABLED	
	ALT5	GPIO2_IO29	ODE - DISABLED	
	ALT6	ARM_TRACE09	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG09	DSE - 40_OHM SRE - SLOW	
LCD_DAT10	ALT0	LCD_DATA10	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA10
	ALT1	KEY_COL1	HYS - ENABLED	
	ALT2	CSI_DATA07	PUS - 100K_OHM_PD	
	ALT3	EIM_DATA04	PUE - KEEP	
	ALT4	ECSPi2_MISO	PKE - ENABLED	
	ALT5	GPIO2_IO30	ODE - DISABLED	
	ALT6	ARM_TRACE10	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG10	DSE - 40_OHM SRE - SLOW	
LCD_DAT11	ALT0	LCD_DATA11	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA11
	ALT1	KEY_ROW1	HYS - ENABLED	
	ALT2	CSI_DATA06	PUS - 100K_OHM_PD	
	ALT3	EIM_DATA05	PUE - KEEP	
	ALT4	ECSPi2_SS1	PKE - ENABLED	
	ALT5	GPIO2_IO31	ODE - DISABLED	
	ALT6	ARM_TRACE11	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG11	DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
LCD_DAT12	ALT0	LCD_DATA12	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA12
	ALT1	KEY_COL2	HYS - ENABLED	
	ALT2	CSI_DATA05	PUS - 100K_OHM_PD	
	ALT3	EIM_DATA06	PUE - KEEP	
	ALT4	UART5_RTS_B	PKE - ENABLED	
	ALT5	GPIO3_IO00	ODE - DISABLED	
	ALT6	ARM_TRACE12	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG12	DSE - 40_OHM SRE - SLOW	
LCD_DAT13	ALT0	LCD_DATA13	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA13
	ALT1	KEY_ROW2	HYS - ENABLED	
	ALT2	CSI_DATA04	PUS - 100K_OHM_PD	
	ALT3	EIM_DATA07	PUE - KEEP	
	ALT4	UART5_CTS_B	PKE - ENABLED	
	ALT5	GPIO3_IO01	ODE - DISABLED	
	ALT6	ARM_TRACE13	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG13	DSE - 40_OHM SRE - SLOW	
LCD_DAT14	ALT0	LCD_DATA14	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA14
	ALT1	KEY_COL3	HYS - ENABLED	
	ALT2	CSI_DATA03	PUS - 100K_OHM_PD	
	ALT3	EIM_DATA08	PUE - KEEP	
	ALT4	UART5_RX_DATA	PKE - ENABLED	
	ALT5	GPIO3_IO02	ODE - DISABLED	
	ALT6	ARM_TRACE14	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG14	DSE - 40_OHM SRE - SLOW	
LCD_DAT15	ALT0	LCD_DATA15	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA15
	ALT1	KEY_ROW3	HYS - ENABLED	
	ALT2	CSI_DATA02	PUS - 100K_OHM_PD	
	ALT3	EIM_DATA09	PUE - KEEP	
	ALT4	UART5_TX_DATA	PKE - ENABLED	
	ALT5	GPIO3_IO03	ODE - DISABLED	
	ALT6	ARM_TRACE15	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG15	DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
LCD_DAT16	ALT0	LCD_DATA16	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA16
	ALT1	KEY_COL4	HYS - ENABLED	
	ALT2	CSI_DATA01	PUS - 100K_OHM_PD	
	ALT3	EIM_DATA10	PUE - KEEP	
	ALT4	I2C2_SCL	PKE - ENABLED	
	ALT5	GPIO3_IO04	ODE - DISABLED	
	ALT6	ARM_TRACE16	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG24	DSE - 40_OHM SRE - SLOW	
LCD_DAT17	ALT0	LCD_DATA17	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA17
	ALT1	KEY_ROW4	HYS - ENABLED	
	ALT2	CSI_DATA00	PUS - 100K_OHM_PD	
	ALT3	EIM_DATA11	PUE - KEEP	
	ALT4	I2C2_SDA	PKE - ENABLED	
	ALT5	GPIO3_IO05	ODE - DISABLED	
	ALT6	ARM_TRACE17	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG25	DSE - 40_OHM SRE - SLOW	
LCD_DAT18	ALT0	LCD_DATA18	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA18
	ALT1	KEY_COL5	HYS - ENABLED	
	ALT2	CSI_DATA15	PUS - 100K_OHM_PD	
	ALT3	EIM_DATA12	PUE - KEEP	
	ALT4	GPT_CAPTURE1	PKE - ENABLED	
	ALT5	GPIO3_IO06	ODE - DISABLED	
	ALT6	ARM_TRACE18	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG26	DSE - 40_OHM SRE - SLOW	
LCD_DAT19	ALT0	LCD_DATA19	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA19
	ALT1	KEY_ROW5	HYS - ENABLED	
	ALT2	CSI_DATA14	PUS - 100K_OHM_PD	
	ALT3	EIM_DATA13	PUE - KEEP	
	ALT4	GPT_CAPTURE2	PKE - ENABLED	
	ALT5	GPIO3_IO07	ODE - DISABLED	
	ALT6	ARM_TRACE19	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG27	DSE - 40_OHM SRE - SLOW	

Table continues on the next page...



**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
LCD_DAT20	ALT0	LCD_DATA20	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA20
	ALT1	KEY_COL6	HYS - ENABLED	
	ALT2	CSI_DATA13	PUS - 100K_OHM_PD	
	ALT3	EIM_DATA14	PUE - KEEP	
	ALT4	GPT_COMPARE1	PKE - ENABLED	
	ALT5	GPIO3_IO08	ODE - DISABLED	
	ALT6	ARM_TRACE20	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG28	DSE - 40_OHM SRE - SLOW	
LCD_DAT21	ALT0	LCD_DATA21	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA21
	ALT1	KEY_ROW6	HYS - ENABLED	
	ALT2	CSI_DATA12	PUS - 100K_OHM_PD	
	ALT3	EIM_DATA15	PUE - KEEP	
	ALT4	GPT_COMPARE2	PKE - ENABLED	
	ALT5	GPIO3_IO09	ODE - DISABLED	
	ALT6	ARM_TRACE21	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG29	DSE - 40_OHM SRE - SLOW	
LCD_DAT22	ALT0	LCD_DATA22	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA22
	ALT1	KEY_COL7	HYS - ENABLED	
	ALT2	CSI_DATA11	PUS - 100K_OHM_PD	
	ALT3	EIM_EB3_B	PUE - KEEP	
	ALT4	GPT_COMPARE3	PKE - ENABLED	
	ALT5	GPIO3_IO10	ODE - DISABLED	
	ALT6	ARM_TRACE22	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG30	DSE - 40_OHM SRE - SLOW	
LCD_DAT23	ALT0	LCD_DATA23	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_DATA23
	ALT1	KEY_ROW7	HYS - ENABLED	
	ALT2	CSI_DATA10	PUS - 100K_OHM_PD	
	ALT3	EIM_EB2_B	PUE - KEEP	
	ALT4	GPT_CLKIN	PKE - ENABLED	
	ALT5	GPIO3_IO11	ODE - DISABLED	
	ALT6	ARM_TRACE23	SPEED - 100MHZ	
	ALT7	SRC_BOOT_CFG31	DSE - 40_OHM SRE - SLOW	

*Table continues on the next page...*

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
LCD_ENABLE	ALT0	LCD_ENABLE	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_ENABLE
	ALT1	SD4_DATA5	HYS - ENABLED	
	ALT2	LCD_RD_E	PUS - 100K_OHM_PD	
	ALT3	EIM_OE_B	PUE - KEEP	
	ALT4	UART2_RX_DATA	PKE - ENABLED	
	ALT5	GPIO2_IO16	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
LCD_HSYNC	ALT0	LCD_HSYNC	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_HSYNC
	ALT1	SD4_DATA6	HYS - ENABLED	
	ALT2	LCD_CS	PUS - 100K_OHM_PD	
	ALT3	EIM_CS0_B	PUE - KEEP	
	ALT4	UART2_TX_DATA	PKE - ENABLED	
	ALT5	GPIO2_IO17	ODE - DISABLED	
	ALT6	ARM_TRACE_CLK	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
LCD_RESET	ALT0	LCD_RESET	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_RESET
	ALT1	EIM_DTACK_B	HYS - ENABLED	
	ALT2	LCD_BUSY	PUS - 100K_OHM_PD	
	ALT3	EIM_WAIT_B	PUE - KEEP	
	ALT4	UART2_CTS_B	PKE - ENABLED	
	ALT5	GPIO2_IO19	ODE - DISABLED	
	ALT6	CCM_PMIC_READY	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
LCD_VSYNC	ALT0	LCD_VSYNC	LVE - DISABLED	SW_PAD_CTL_PAD_LCD_VSYNC
	ALT1	SD4_DATA7	HYS - ENABLED	
	ALT2	LCD_RS	PUS - 100K_OHM_PD	
	ALT3	EIM_CS1_B	PUE - KEEP	
	ALT4	UART2_RTS_B	PKE - ENABLED	
	ALT5	GPIO2_IO18	ODE - DISABLED	
	ALT6	ARM_TRACE_CTL	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
ONOFF		SRC_ONOFF		

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
PMIC_ON_REQ		SNVS_PMIC_ON_REQ		
PMIC_STBY_REQ		CCM_PMIC_STBY_REQ		
POR_B		SRC_POR_B		
PWM1	ALT0	PWM1_OUT	LVE - DISABLED	SW_PAD_CTL_PAD_PWM1
	ALT1	CCM_CLKO	HYS - ENABLED	
	ALT2	AUDIO_CLK_OUT	PUS - 100K_OHM_PD	
	ALT3	FEC_REF_OUT	PUE - KEEP	
	ALT4	CSI_MCLK	PKE - ENABLED	
	ALT5	GPIO3_IO23	ODE - DISABLED	
	ALT6	EPIT1_OUT	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
REF_CLK_24M	ALT0	XTALOSC_REF_CLK_24M	LVE - DISABLED	SW_PAD_CTL_PAD_REF_CLK_24M
	ALT1	I2C3_SCL	HYS - ENABLED	
	ALT2	PWM3_OUT	PUS - 100K_OHM_PD	
	ALT3	USB_OTG2_ID	PUE - KEEP	
	ALT4	CCM_PMIC_READY	PKE - ENABLED	
	ALT5	GPIO3_IO21	ODE - DISABLED	
	ALT6	SD3_WP	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
REF_CLK_32K	ALT0	XTALOSC_REF_CLK_32K	LVE - DISABLED	SW_PAD_CTL_PAD_REF_CLK_32K
	ALT1	I2C3_SDA	HYS - ENABLED	
	ALT2	PWM4_OUT	PUS - 100K_OHM_PD	
	ALT3	USB_OTG1_ID	PUE - KEEP	
	ALT4	SD1_LCTL	PKE - ENABLED	
	ALT5	GPIO3_IO22	ODE - DISABLED	
	ALT6	SD3_CD_B	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
RTC_XTALI		XTALOSC_RTC_XTALI		
RTC_XTALO		XTALOSC_RTC_XTALO		

*Table continues on the next page...*

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
SD1_CLK	ALT0	SD1_CLK	LVE - DISABLED	SW_PAD_CTL_PAD_SD1_CLK
	ALT1	FEC_MDIO	HYS - ENABLED	
	ALT2	KEY_COL0	PUS - 100K_OHM_PD	
	ALT3	EPDC_SDCE4	PUE - KEEP	
	ALT5	GPIO5_IO15	PKE - ENABLED ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD1_CMD	ALT0	SD1_CMD	LVE - DISABLED	SW_PAD_CTL_PAD_SD1_CMD
	ALT1	FEC_TX_CLK	HYS - ENABLED	
	ALT2	KEY_ROW0	PUS - 100K_OHM_PD	
	ALT3	EPDC_SDCE5	PUE - KEEP	
	ALT5	GPIO5_IO14	PKE - ENABLED ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD1_DAT0	ALT0	SD1_DATA0	LVE - DISABLED	SW_PAD_CTL_PAD_SD1_DATA0
	ALT1	FEC_RX_ER	HYS - ENABLED	
	ALT2	KEY_COL1	PUS - 100K_OHM_PD	
	ALT3	EPDC_SDCE6	PUE - KEEP	
	ALT5	GPIO5_IO11	PKE - ENABLED ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD1_DAT1	ALT0	SD1_DATA1	LVE - DISABLED	SW_PAD_CTL_PAD_SD1_DATA1
	ALT1	FEC_RX_DV	HYS - ENABLED	
	ALT2	KEY_ROW1	PUS - 100K_OHM_PD	
	ALT3	EPDC_SDCE7	PUE - KEEP	
	ALT5	GPIO5_IO08	PKE - ENABLED ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
SD1_DAT2	ALT0	SD1_DATA2	LVE - DISABLED	SW_PAD_CTL_PAD_SD1_DATA2
	ALT1	FEC_RX_DATA1	HYS - ENABLED	
	ALT2	KEY_COL2	PUS - 100K_OHM_PD	
	ALT3	EPDC_SDCE8	PUE - KEEP	
	ALT5	GPIO5_IO13	PKE - ENABLED ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD1_DAT3	ALT0	SD1_DATA3	LVE - DISABLED	SW_PAD_CTL_PAD_SD1_DATA3
	ALT1	FEC_TX_DATA0	HYS - ENABLED	
	ALT2	KEY_ROW2	PUS - 100K_OHM_PD	
	ALT3	EPDC_SDCE9	PUE - KEEP	
	ALT5	GPIO5_IO06	PKE - ENABLED ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD1_DAT4	ALT0	SD1_DATA4	LVE - DISABLED	SW_PAD_CTL_PAD_SD1_DATA4
	ALT1	FEC_MDC	HYS - ENABLED	
	ALT2	KEY_COL3	PUS - 100K_OHM_PD	
	ALT3	EPDC_SDCLK_N	PUE - KEEP	
	ALT4	UART4_RX_DATA	PKE - ENABLED	
	ALT5	GPIO5_IO12	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD1_DAT5	ALT0	SD1_DATA5	LVE - DISABLED	SW_PAD_CTL_PAD_SD1_DATA5
	ALT1	FEC_RX_DATA0	HYS - ENABLED	
	ALT2	KEY_ROW3	PUS - 100K_OHM_PD	
	ALT3	EPDC_SDOED	PUE - KEEP	
	ALT4	UART4_TX_DATA	PKE - ENABLED	
	ALT5	GPIO5_IO09	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
SD1_DAT6	ALT0	SD1_DATA6	LVE - DISABLED	SW_PAD_CTL_PAD_SD1_DATA6
	ALT1	FEC_TX_EN	HYS - ENABLED	
	ALT2	KEY_COL4	PUS - 100K_OHM_PD	
	ALT3	EPDC_SDOEZ	PUE - KEEP	
	ALT4	UART4_RTS_B	PKE - ENABLED	
	ALT5	GPIO5_IO07	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD1_DAT7	ALT0	SD1_DATA7	LVE - DISABLED	SW_PAD_CTL_PAD_SD1_DATA7
	ALT1	FEC_TX_DATA1	HYS - ENABLED	
	ALT2	KEY_ROW4	PUS - 100K_OHM_PD	
	ALT3	CCM_PMIC_READY	PUE - KEEP	
	ALT4	UART4_CTS_B	PKE - ENABLED	
	ALT5	GPIO5_IO10	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD2_CLK	ALT0	SD2_CLK	LVE - DISABLED	SW_PAD_CTL_PAD_SD2_CLK
	ALT1	AUD4_RXFS	HYS - ENABLED	
	ALT2	ECSPI3_SCLK	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA00	PUE - KEEP	
	ALT5	GPIO5_IO05	PKE - ENABLED ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD2_CMD	ALT0	SD2_CMD	LVE - DISABLED	SW_PAD_CTL_PAD_SD2_CMD
	ALT1	AUD4_RXC	HYS - ENABLED	
	ALT2	ECSPI3_SS0	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA01	PUE - KEEP	
	ALT4	EPIT1_OUT	PKE - ENABLED	
	ALT5	GPIO5_IO04	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
SD2_DAT0	ALT0	SD2_DATA0	LVE - DISABLED	SW_PAD_CTL_PAD_SD2_DATA0
	ALT1	AUD4_RXD	HYS - ENABLED	
	ALT2	ECSPI3_MOSI	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA02	PUE - KEEP	
	ALT4	UART5_RTS_B	PKE - ENABLED	
	ALT5	GPIO5_IO01	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD2_DAT1	ALT0	SD2_DATA1	LVE - DISABLED	SW_PAD_CTL_PAD_SD2_DATA1
	ALT1	AUD4_TXC	HYS - ENABLED	
	ALT2	ECSPI3_MISO	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA03	PUE - KEEP	
	ALT4	UART5_CTS_B	PKE - ENABLED	
	ALT5	GPIO4_IO30	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD2_DAT2	ALT0	SD2_DATA2	LVE - DISABLED	SW_PAD_CTL_PAD_SD2_DATA2
	ALT1	AUD4_TXFS	HYS - ENABLED	
	ALT2	FEC_COL	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA04	PUE - KEEP	
	ALT4	UART5_RX_DATA	PKE - ENABLED	
	ALT5	GPIO5_IO03	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD2_DAT3	ALT0	SD2_DATA3	LVE - DISABLED	SW_PAD_CTL_PAD_SD2_DATA3
	ALT1	AUD4_TXD	HYS - ENABLED	
	ALT2	FEC_RX_CLK	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA05	PUE - KEEP	
	ALT4	UART5_TX_DATA	PKE - ENABLED	
	ALT5	GPIO4_IO28	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
SD2_DAT4	ALT0	SD2_DATA4	LVE - DISABLED	SW_PAD_CTL_PAD_SD2_DATA4
	ALT1	SD3_DATA4	HYS - ENABLED	
	ALT2	UART2_RX_DATA	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA06	PUE - KEEP	
	ALT4	SPDIF_OUT	PKE - ENABLED	
	ALT5	GPIO5_IO02	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD2_DAT5	ALT0	SD2_DATA5	LVE - DISABLED	SW_PAD_CTL_PAD_SD2_DATA5
	ALT1	SD3_DATA5	HYS - ENABLED	
	ALT2	UART2_TX_DATA	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA07	PUE - KEEP	
	ALT4	SPDIF_IN	PKE - ENABLED	
	ALT5	GPIO4_IO31	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD2_DAT6	ALT0	SD2_DATA6	LVE - DISABLED	SW_PAD_CTL_PAD_SD2_DATA6
	ALT1	SD3_DATA6	HYS - ENABLED	
	ALT2	UART2_RTS_B	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA08	PUE - KEEP	
	ALT4	SD2_WP	PKE - ENABLED	
	ALT5	GPIO4_IO29	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD2_DAT7	ALT0	SD2_DATA7	LVE - DISABLED	SW_PAD_CTL_PAD_SD2_DATA7
	ALT1	SD3_DATA7	HYS - ENABLED	
	ALT2	UART2_CTS_B	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA09	PUE - KEEP	
	ALT4	SD2_CD_B	PKE - ENABLED	
	ALT5	GPIO5_IO00	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...



**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
SD2_RST	ALT0	SD2_RESET	LVE - DISABLED	SW_PAD_CTL_PAD_SD2_RESET
	ALT1	FEC_REF_OUT	HYS - ENABLED	
	ALT2	WDOG2_B	PUS - 100K_OHM_PD	
	ALT3	SPDIF_OUT	PUE - KEEP	
	ALT4	CSI_MCLK	PKE - ENABLED	
	ALT5	GPIO4_IO27	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD3_CLK	ALT0	SD3_CLK	LVE - DISABLED	SW_PAD_CTL_PAD_SD3_CLK
	ALT1	AUD5_RXFS	HYS - ENABLED	
	ALT2	KEY_COL5	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA10	PUE - KEEP	
	ALT4	WDOG1_RESET_B_DEB	PKE - ENABLED	
	ALT5	GPIO5_IO18	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD3_CMD	ALT0	SD3_CMD	LVE - DISABLED	SW_PAD_CTL_PAD_SD3_CMD
	ALT1	AUD5_RXC	HYS - ENABLED	
	ALT2	KEY_ROW5	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA11	PUE - KEEP	
	ALT4	USB_OTG2_ID	PKE - ENABLED	
	ALT5	GPIO5_IO21	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD3_DATA0	ALT0	SD3_DATA0	LVE - DISABLED	SW_PAD_CTL_PAD_SD3_DATA0
	ALT1	AUD5_RXD	HYS - ENABLED	
	ALT2	KEY_COL6	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA12	PUE - KEEP	
	ALT4	USB_OTG1_ID	PKE - ENABLED	
	ALT5	GPIO5_IO19	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
SD3_DAT1	ALT0	SD3_DATA1	LVE - DISABLED	SW_PAD_CTL_PAD_SD3_DATA1
	ALT1	AUD5_TXC	HYS - ENABLED	
	ALT2	KEY_ROW6	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA13	PUE - KEEP	
	ALT4	SD1_VSELECT	PKE - ENABLED	
	ALT5	GPIO5_IO20	ODE - DISABLED	
	ALT6	JTAG_DE_B	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD3_DAT2	ALT0	SD3_DATA2	LVE - DISABLED	SW_PAD_CTL_PAD_SD3_DATA2
	ALT1	AUD5_TXFS	HYS - ENABLED	
	ALT2	KEY_COL7	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA14	PUE - KEEP	
	ALT4	EPIT1_OUT	PKE - ENABLED	
	ALT5	GPIO5_IO16	ODE - DISABLED	
	ALT6	USB_OTG2_OC	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
SD3_DAT3	ALT0	SD3_DATA3	LVE - DISABLED	SW_PAD_CTL_PAD_SD3_DATA3
	ALT1	AUD5_TXD	HYS - ENABLED	
	ALT2	KEY_ROW7	PUS - 100K_OHM_PD	
	ALT3	CSI_DATA15	PUE - KEEP	
	ALT4	EPIT2_OUT	PKE - ENABLED	
	ALT5	GPIO5_IO17	ODE - DISABLED	
	ALT6	USB_OTG1_OC	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
TAMPER		SNVS_TAMPER		
TEST_MODE		TCU_TEST_MODE		

*Table continues on the next page...*

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
UART1_RXD	ALT0	UART1_RX_DATA	LVE - DISABLED	SW_PAD_CTL_PAD_UART1_RXD
	ALT1	PWM1_OUT	HYS - ENABLED	
	ALT2	UART4_RX_DATA	PUS - 100K_OHM_PD	
	ALT3	FEC_COL	PUE - KEEP	
	ALT4	UART5_RX_DATA	PKE - ENABLED	
	ALT5	GPIO3_IO16	ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
UART1_TXD	ALT0	UART1_TX_DATA	LVE - DISABLED	SW_PAD_CTL_PAD_UART1_TXD
	ALT1	PWM2_OUT	HYS - ENABLED	
	ALT2	UART4_TX_DATA	PUS - 100K_OHM_PD	
	ALT3	FEC_RX_CLK	PUE - KEEP	
	ALT4	UART5_TX_DATA	PKE - ENABLED	
	ALT5	GPIO3_IO17	ODE - DISABLED	
	ALT7	UART5_DCD_B	SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
HSIC_DAT	ALT0	USB_H_DATA	DDR_SEL - RESERVED0	SW_PAD_CTL_PAD_USB_H_DATA
	ALT1	I2C1_SCL	DDR_INPUT - CMOS	
	ALT2	PWM1_OUT	HYS - DISABLED	
	ALT3	XTALOSC_REF_CLK_24M	PUS - 100K_OHM_PD	
	ALT5	GPIO3_IO19	PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	
HSIC_STROBE	ALT0	USB_H_STROBE	DDR_SEL - RESERVED0	SW_PAD_CTL_PAD_USB_H_STROBE
	ALT1	I2C1_SDA	DDR_INPUT - CMOS	
	ALT2	PWM2_OUT	HYS - DISABLED	
	ALT3	XTALOSC_REF_CLK_32K	PUS - 100K_OHM_PD	
	ALT5	GPIO3_IO20	PUE - PULL PKE - ENABLED ODT - DISABLED DSE - 40_OHM	
USB_OTG1_DN		USB_OTG1_DN		
USB_OTG1_DP		USB_OTG1_DP		

Table continues on the next page...

**Table 4-1. Pin Assignments (continued)**

Pad Name	Mode	Signal	Pad Settings	Pad/Group Registers
USB_OTG2_DN		USB_OTG2_DN		
USB_OTG2_DP		USB_OTG2_DP		
USB_OTG_CHD_B		USB_OTG_CHD_B		
WDOG_B	ALT0	WDOG1_B	LVE - DISABLED	SW_PAD_CTL_PAD_WDOG_B
	ALT1	WDOG1_RESET_B_DEB	HYS - ENABLED	
	ALT2	UART5_RI_B	PUS - 100K_OHM_PD	
	ALT5	GPIO3_IO18	PUE - KEEP PKE - ENABLED ODE - DISABLED SPEED - 100MHZ DSE - 40_OHM SRE - SLOW	
XTALI		XTALOSC_XTALI		
XTALO		XTALOSC_XTALO		

## 4.1.2 Muxing Options

An additional view of external signals muxing is shown by the presentation of the muxing options per block/instance.

**Table 4-2. Muxing Options**

Signal	Pad (Mode)	Mux/Input Select Registers
ARM - ARM Platform		
ARM_EVENTI	EPDC_PWRSTAT (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_STAT
ARM_EVENTO	EPDC_PWRWAKEUP (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_WAKE
ARM_TRACE00	LCD_DAT0 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA00
ARM_TRACE01	LCD_DAT1 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA01
ARM_TRACE02	LCD_DAT2 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA02
ARM_TRACE03	LCD_DAT3 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA03
ARM_TRACE04	LCD_DAT4 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA04
ARM_TRACE05	LCD_DAT5 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA05
ARM_TRACE06	LCD_DAT6 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA06
ARM_TRACE07	LCD_DAT7 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA07
ARM_TRACE08	LCD_DAT8 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA08
ARM_TRACE09	LCD_DAT9 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA09

*Table continues on the next page...*

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
ARM_TRACE10	LCD_DAT10 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA10
ARM_TRACE11	LCD_DAT11 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA11
ARM_TRACE12	LCD_DAT12 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA12
ARM_TRACE13	LCD_DAT13 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA13
ARM_TRACE14	LCD_DAT14 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA14
ARM_TRACE15	LCD_DAT15 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA15
ARM_TRACE16	LCD_DAT16 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA16
ARM_TRACE17	LCD_DAT17 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA17
ARM_TRACE18	LCD_DAT18 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA18
ARM_TRACE19	LCD_DAT19 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA19
ARM_TRACE20	LCD_DAT20 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA20
ARM_TRACE21	LCD_DAT21 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA21
ARM_TRACE22	LCD_DAT22 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA22
ARM_TRACE23	LCD_DAT23 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA23
ARM_TRACE24	FEC_RXD0 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA0
ARM_TRACE25	FEC_RX_ER (ALT6)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_ER
ARM_TRACE26	FEC_MDIO (ALT6)	IOMUXC_SW_MUX_CTL_PAD_FEC_MDIO
ARM_TRACE27	FEC_TX_CLK (ALT6)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_CLK
ARM_TRACE28	FEC_TX_EN (ALT6)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_EN
ARM_TRACE29	FEC_MDC (ALT6)	IOMUXC_SW_MUX_CTL_PAD_FEC_MDC
ARM_TRACE30	FEC_TXD0 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA0
ARM_TRACE31	FEC_CRD_DV (ALT6)	IOMUXC_SW_MUX_CTL_PAD_FEC_CRD_DV
ARM_TRACE_CLK	LCD_HSYNC (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_HSYNC
ARM_TRACE_CTL	LCD_VSYNC (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_VSYNC
<b>AUDMUX - Digital Audio Multiplexer</b>		
AUD3_RXC	AUD_RXC (ALT0)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXC
AUD3_RXD	AUD_RXD (ALT0)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXD
AUD3_RXFS	AUD_RXFS (ALT0)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXFS
AUD3_TXC	AUD_TXC (ALT0)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXC
AUD3_TXD	AUD_TXD (ALT0)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXD
AUD3_TXFS	AUD_TXFS (ALT0)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXFS
AUD4_RXC	I2C2_SDA (ALT1)	IOMUXC_SW_MUX_CTL_PAD_I2C2_SDA IOMUXC_AUD4_INPUT_RXCLK_AMX_SELECT_INPUT
	LCD_DAT2 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA02 IOMUXC_AUD4_INPUT_RXCLK_AMX_SELECT_INPUT
	SD2_CMD (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD2_CMD IOMUXC_AUD4_INPUT_RXCLK_AMX_SELECT_INPUT

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
AUD4_RXD	ECSPI1_SS0 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_ECSPI1_SS0 IOMUXC_AUD4_INPUT_DA_AMX_SELECT_INPUT
	LCD_DAT3 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA03 IOMUXC_AUD4_INPUT_DA_AMX_SELECT_INPUT
	SD2_DAT0 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA0 IOMUXC_AUD4_INPUT_DA_AMX_SELECT_INPUT
AUD4_RXFS	I2C2_SCL (ALT1)	IOMUXC_SW_MUX_CTL_PAD_I2C2_SCL IOMUXC_AUD4_INPUT_RXFS_AMX_SELECT_INPUT
	LCD_DAT1 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA01 IOMUXC_AUD4_INPUT_RXFS_AMX_SELECT_INPUT
	SD2_CLK (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD2_CLK IOMUXC_AUD4_INPUT_RXFS_AMX_SELECT_INPUT
AUD4_TXC	ECSPI1_MOSI (ALT1)	IOMUXC_SW_MUX_CTL_PAD_ECSPI1_MOSI IOMUXC_AUD4_INPUT_TXCLK_AMX_SELECT_INPUT
	LCD_DAT4 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA04 IOMUXC_AUD4_INPUT_TXCLK_AMX_SELECT_INPUT
	SD2_DAT1 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA1 IOMUXC_AUD4_INPUT_TXCLK_AMX_SELECT_INPUT
AUD4_TXD	ECSPI1_SCLK (ALT1)	IOMUXC_SW_MUX_CTL_PAD_ECSPI1_SCLK IOMUXC_AUD4_INPUT_DB_AMX_SELECT_INPUT
	LCD_DAT6 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA06 IOMUXC_AUD4_INPUT_DB_AMX_SELECT_INPUT
	SD2_DAT3 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA3 IOMUXC_AUD4_INPUT_DB_AMX_SELECT_INPUT
AUD4_TXFS	ECSPI1_MISO (ALT1)	IOMUXC_SW_MUX_CTL_PAD_ECSPI1_MISO IOMUXC_AUD4_INPUT_TXFS_AMX_SELECT_INPUT
	LCD_DAT5 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA05 IOMUXC_AUD4_INPUT_TXFS_AMX_SELECT_INPUT
	SD2_DAT2 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA2 IOMUXC_AUD4_INPUT_TXFS_AMX_SELECT_INPUT
AUD5_RXC	EPDC_PWRCTRL0 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL0 IOMUXC_AUD5_INPUT_RXCLK_AMX_SELECT_INPUT
	SD3_CMD (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD3_CMD IOMUXC_AUD5_INPUT_RXCLK_AMX_SELECT_INPUT
AUD5_RXD	EPDC_VCOM1 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM1 IOMUXC_AUD5_INPUT_DA_AMX_SELECT_INPUT
	SD3_DAT0 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA0 IOMUXC_AUD5_INPUT_DA_AMX_SELECT_INPUT

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
AUD5_RXFS	EPDC_VCOM0 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM0 IOMUXC_AUD5_INPUT_RXFS_AMX_SELECT_INPUT
	SD3_CLK (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD3_CLK IOMUXC_AUD5_INPUT_RXFS_AMX_SELECT_INPUT
AUD5_TXC	EPDC_PWRCTRL3 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL3 IOMUXC_AUD5_INPUT_TXCLK_AMX_SELECT_INPUT
	SD3_DAT1 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA1 IOMUXC_AUD5_INPUT_TXCLK_AMX_SELECT_INPUT
AUD5_TXD	EPDC_PWRCTRL2 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL2 IOMUXC_AUD5_INPUT_DB_AMX_SELECT_INPUT
	SD3_DAT3 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA3 IOMUXC_AUD5_INPUT_DB_AMX_SELECT_INPUT
AUD5_TXFS	EPDC_PWRCTRL1 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL1 IOMUXC_AUD5_INPUT_TXFS_AMX_SELECT_INPUT
	SD3_DAT2 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA2 IOMUXC_AUD5_INPUT_TXFS_AMX_SELECT_INPUT
AUD6_RXC	FEC_TX_CLK (ALT2)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_CLK IOMUXC_AUD6_INPUT_RXCLK_AMX_SELECT_INPUT
	KEY_ROW3 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW3 IOMUXC_AUD6_INPUT_RXCLK_AMX_SELECT_INPUT
AUD6_RXD	FEC_RX_ER (ALT2)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_ER IOMUXC_AUD6_INPUT_DA_AMX_SELECT_INPUT
	KEY_COL4 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL4 IOMUXC_AUD6_INPUT_DA_AMX_SELECT_INPUT
AUD6_RXFS	FEC_MDIO (ALT2)	IOMUXC_SW_MUX_CTL_PAD_FEC_MDIO IOMUXC_AUD6_INPUT_RXFS_AMX_SELECT_INPUT
	KEY_COL3 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL3 IOMUXC_AUD6_INPUT_RXFS_AMX_SELECT_INPUT
AUD6_TXC	FEC_CRS_DV (ALT2)	IOMUXC_SW_MUX_CTL_PAD_FEC_CRS_DV IOMUXC_AUD6_INPUT_TXCLK_AMX_SELECT_INPUT
	KEY_ROW4 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW4 IOMUXC_AUD6_INPUT_TXCLK_AMX_SELECT_INPUT
AUD6_TXD	FEC_TXD0 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA0 IOMUXC_AUD6_INPUT_DB_AMX_SELECT_INPUT
	KEY_ROW5 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW5 IOMUXC_AUD6_INPUT_DB_AMX_SELECT_INPUT

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
AUD6_TXFS	FEC_RXD1 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA1 IOMUXC_AUD6_INPUT_TXFS_AMX_SELECT_INPUT
	KEY_COL5 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL5 IOMUXC_AUD6_INPUT_TXFS_AMX_SELECT_INPUT
AUDIO_CLK_OUT	AUD_MCLK (ALT0)	IOMUXC_SW_MUX_CTL_PAD_AUD_MCLK
	FEC_MDC (ALT2)	IOMUXC_SW_MUX_CTL_PAD_FEC_MDC
	LCD_DAT7 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA07
	PWM1 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_PWM1
CCM - Clock Controller Module		
CCM_CLKO	PWM1 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_PWM1
CCM_PMIC_READY	FEC_REF_CLK (ALT4)	IOMUXC_SW_MUX_CTL_PAD_FEC_REF_CLK IOMUXC_CCM_PMIC_READY_SELECT_INPUT
	LCD_RESET (ALT6)	IOMUXC_SW_MUX_CTL_PAD_LCD_RESET IOMUXC_CCM_PMIC_READY_SELECT_INPUT
	REF_CLK_24M (ALT4)	IOMUXC_SW_MUX_CTL_PAD_REF_CLK_24M IOMUXC_CCM_PMIC_READY_SELECT_INPUT
	SD1_DAT7 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA7 IOMUXC_CCM_PMIC_READY_SELECT_INPUT
CCM_PMIC_STBY_REQ	PMIC_STBY_REQ	Not multiplexed.
CSI - CMOS Sensor Interface		
CSI_DATA00	EPDC_D0 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA00 IOMUXC_CSI_CSI_DATA00_SELECT_INPUT
	LCD_DAT17 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA17 IOMUXC_CSI_CSI_DATA00_SELECT_INPUT
	SD2_CLK (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD2_CLK IOMUXC_CSI_CSI_DATA00_SELECT_INPUT
CSI_DATA01	EPDC_D1 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA01 IOMUXC_CSI_CSI_DATA01_SELECT_INPUT
	LCD_DAT16 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA16 IOMUXC_CSI_CSI_DATA01_SELECT_INPUT
	SD2_CMD (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD2_CMD IOMUXC_CSI_CSI_DATA01_SELECT_INPUT
CSI_DATA02	EPDC_D2 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA02 IOMUXC_CSI_CSI_DATA02_SELECT_INPUT
	LCD_DAT15 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA15 IOMUXC_CSI_CSI_DATA02_SELECT_INPUT
	SD2_DAT0 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA0 IOMUXC_CSI_CSI_DATA02_SELECT_INPUT

Table continues on the next page...



**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
CSI_DATA03	EPDC_D3 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA03 IOMUXC_CSI_CSI_DATA03_SELECT_INPUT
	LCD_DAT14 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA14 IOMUXC_CSI_CSI_DATA03_SELECT_INPUT
	SD2_DAT1 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA1 IOMUXC_CSI_CSI_DATA03_SELECT_INPUT
CSI_DATA04	EPDC_D4 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA04 IOMUXC_CSI_CSI_DATA04_SELECT_INPUT
	LCD_DAT13 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA13 IOMUXC_CSI_CSI_DATA04_SELECT_INPUT
	SD2_DAT2 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA2 IOMUXC_CSI_CSI_DATA04_SELECT_INPUT
CSI_DATA05	EPDC_D5 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA05 IOMUXC_CSI_CSI_DATA05_SELECT_INPUT
	LCD_DAT12 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA12 IOMUXC_CSI_CSI_DATA05_SELECT_INPUT
	SD2_DAT3 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA3 IOMUXC_CSI_CSI_DATA05_SELECT_INPUT
CSI_DATA06	EPDC_D6 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA06 IOMUXC_CSI_CSI_DATA06_SELECT_INPUT
	LCD_DAT11 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA11 IOMUXC_CSI_CSI_DATA06_SELECT_INPUT
	SD2_DAT4 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA4 IOMUXC_CSI_CSI_DATA06_SELECT_INPUT
CSI_DATA07	EPDC_D7 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA07 IOMUXC_CSI_CSI_DATA07_SELECT_INPUT
	LCD_DAT10 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA10 IOMUXC_CSI_CSI_DATA07_SELECT_INPUT
	SD2_DAT5 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA5 IOMUXC_CSI_CSI_DATA07_SELECT_INPUT
CSI_DATA08	EPDC_SDCLK (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCLK IOMUXC_CSI_CSI_DATA08_SELECT_INPUT
	LCD_DAT9 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA09 IOMUXC_CSI_CSI_DATA08_SELECT_INPUT
	SD2_DAT6 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA6 IOMUXC_CSI_CSI_DATA08_SELECT_INPUT

*Table continues on the next page...*

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
CSI_DATA09	EPDC_SDLE (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDLE IOMUXC_CSI_CSI_DATA09_SELECT_INPUT
	LCD_DAT8 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA08 IOMUXC_CSI_CSI_DATA09_SELECT_INPUT
	SD2_DAT7 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA7 IOMUXC_CSI_CSI_DATA09_SELECT_INPUT
CSI_DATA10	EPDC_SDOE (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDOE IOMUXC_CSI_CSI_DATA10_SELECT_INPUT
	LCD_DAT23 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA23 IOMUXC_CSI_CSI_DATA10_SELECT_INPUT
	SD3_CLK (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD3_CLK IOMUXC_CSI_CSI_DATA10_SELECT_INPUT
CSI_DATA11	EPDC_SDSHR (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDSHR IOMUXC_CSI_CSI_DATA11_SELECT_INPUT
	LCD_DAT22 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA22 IOMUXC_CSI_CSI_DATA11_SELECT_INPUT
	SD3_CMD (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD3_CMD IOMUXC_CSI_CSI_DATA11_SELECT_INPUT
CSI_DATA12	LCD_DAT21 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA21 IOMUXC_CSI_CSI_DATA12_SELECT_INPUT
	SD3_DAT0 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA0 IOMUXC_CSI_CSI_DATA12_SELECT_INPUT
CSI_DATA13	LCD_DAT20 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA20 IOMUXC_CSI_CSI_DATA13_SELECT_INPUT
	SD3_DAT1 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA1 IOMUXC_CSI_CSI_DATA13_SELECT_INPUT
CSI_DATA14	LCD_DAT19 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA19 IOMUXC_CSI_CSI_DATA14_SELECT_INPUT
	SD3_DAT2 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA2 IOMUXC_CSI_CSI_DATA14_SELECT_INPUT
CSI_DATA15	LCD_DAT18 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA18 IOMUXC_CSI_CSI_DATA15_SELECT_INPUT
	SD3_DAT3 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA3 IOMUXC_CSI_CSI_DATA15_SELECT_INPUT

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
CSI_HSYNC	ECSPI2_MOSI (ALT3)	IOMUXC_SW_MUX_CTL_PAD_ECSPi2_MOSI IOMUXC_CSI_CSI_HSYNC_SELECT_INPUT
	EPDC_GDOE (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDOE IOMUXC_CSI_CSI_HSYNC_SELECT_INPUT
	LCD_DAT5 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA05 IOMUXC_CSI_CSI_HSYNC_SELECT_INPUT
CSI_MCLK	ECSPI2_MISO (ALT3)	IOMUXC_SW_MUX_CTL_PAD_ECSPi2_MISO
	EPDC_GDRL (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDRL
	LCD_DAT7 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA07
	PWM1 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_PWM1
	SD2_RST (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD2_RESET
CSI_PIXCLK	ECSPI2_SCLK (ALT3)	IOMUXC_SW_MUX_CTL_PAD_ECSPi2_SCLK IOMUXC_CSI_CSI_PIXCLK_SELECT_INPUT
	EPDC_GDCLK (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDCLK IOMUXC_CSI_CSI_PIXCLK_SELECT_INPUT
	LCD_DAT6 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA06 IOMUXC_CSI_CSI_PIXCLK_SELECT_INPUT
CSI_VSYNC	ECSPI2_SS0 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_ECSPi2_SS0 IOMUXC_CSI_CSI_VSYNC_SELECT_INPUT
	EPDC_GDSP (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDSP IOMUXC_CSI_CSI_VSYNC_SELECT_INPUT
	LCD_DAT4 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA04 IOMUXC_CSI_CSI_VSYNC_SELECT_INPUT
<b>ECSPi1 - Enhanced Configurable SPI</b>		
ECSPi1_MISO	ECSPi1_MISO (ALT0)	IOMUXC_SW_MUX_CTL_PAD_ECSPi1_MISO IOMUXC_ECSPi1_MISO_SELECT_INPUT
	LCD_DAT1 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA01 IOMUXC_ECSPi1_MISO_SELECT_INPUT
ECSPi1_MOSI	ECSPi1_MOSI (ALT0)	IOMUXC_SW_MUX_CTL_PAD_ECSPi1_MOSI IOMUXC_ECSPi1_MOSI_SELECT_INPUT
	LCD_DAT0 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA00 IOMUXC_ECSPi1_MOSI_SELECT_INPUT
ECSPi1_RDY	I2C2_SCL (ALT6)	IOMUXC_SW_MUX_CTL_PAD_I2C2_SCL IOMUXC_ECSPi1_DATAREADY_B_SELECT_INPUT
	LCD_DAT7 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA07 IOMUXC_ECSPi1_DATAREADY_B_SELECT_INPUT

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
ECSPI1_SCLK	ECSPI1_SCLK (ALT0)	IOMUXC_SW_MUX_CTL_PAD_ECSP11_SCLK IOMUXC_ECSP11_CSPI_CLK_IN_SELECT_INPUT
	LCD_DAT3 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA03 IOMUXC_ECSP11_CSPI_CLK_IN_SELECT_INPUT
ECSPI1_SS0	ECSPI1_SS0 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_ECSP11_SS0 IOMUXC_ECSP11_SS0_SELECT_INPUT
	LCD_DAT2 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA02 IOMUXC_ECSP11_SS0_SELECT_INPUT
ECSPI1_SS1	I2C1_SCL (ALT6)	IOMUXC_SW_MUX_CTL_PAD_I2C1_SCL IOMUXC_ECSP11_SS1_SELECT_INPUT
	LCD_DAT4 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA04 IOMUXC_ECSP11_SS1_SELECT_INPUT
ECSPI1_SS2	I2C1_SDA (ALT6)	IOMUXC_SW_MUX_CTL_PAD_I2C1_SDA IOMUXC_ECSP11_SS2_SELECT_INPUT
	LCD_DAT5 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA05 IOMUXC_ECSP11_SS2_SELECT_INPUT
ECSPI1_SS3	ECSPI2_SS0 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_ECSP12_SS0 IOMUXC_ECSP11_SS3_SELECT_INPUT
	LCD_DAT6 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA06 IOMUXC_ECSP11_SS3_SELECT_INPUT
<b>ECSPI2 - Enhanced Configurable SPI</b>		
ECSPI2_MISO	ECSPI2_MISO (ALT0)	IOMUXC_SW_MUX_CTL_PAD_ECSP12_MISO IOMUXC_ECSP12_MISO_SELECT_INPUT
	EPDC_SDLE (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDLE IOMUXC_ECSP12_MISO_SELECT_INPUT
	LCD_DAT10 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA10 IOMUXC_ECSP12_MISO_SELECT_INPUT
ECSPI2_MOSI	ECSPI2_MOSI (ALT0)	IOMUXC_SW_MUX_CTL_PAD_ECSP12_MOSI IOMUXC_ECSP12_MOSI_SELECT_INPUT
	EPDC_SDCLK (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCLK IOMUXC_ECSP12_MOSI_SELECT_INPUT
	LCD_DAT9 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA09 IOMUXC_ECSP12_MOSI_SELECT_INPUT
ECSPI2_RDY	EPDC_GDRL (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDRL

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
ECSPI2_SCLK	ECSPI2_SCLK (ALT0)	IOMUXC_SW_MUX_CTL_PAD_ECSPi2_SCLK IOMUXC_ECSPi2_CSPI_CLK_IN_SELECT_INPUT
	EPDC_SDSHR (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDSHR IOMUXC_ECSPi2_CSPI_CLK_IN_SELECT_INPUT
	LCD_DAT8 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA08 IOMUXC_ECSPi2_CSPI_CLK_IN_SELECT_INPUT
ECSPI2_SS0	ECSPI2_SS0 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_ECSPi2_SS0 IOMUXC_ECSPi2_SS0_SELECT_INPUT
	EPDC_SDOE (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDOE IOMUXC_ECSPi2_SS0_SELECT_INPUT
ECSPI2_SS1	EPDC_SDCE0 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE0 IOMUXC_ECSPi2_SS1_SELECT_INPUT
	LCD_DAT11 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA11 IOMUXC_ECSPi2_SS1_SELECT_INPUT
ECSPI2_SS2	EPDC_GDCLK (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDCLK
ECSPI2_SS3	EPDC_GDOE (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDOE
<b>ECSPI3 - Enhanced Configurable SPI</b>		
ECSPI3_MISO	AUD_TXC (ALT1)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXC IOMUXC_ECSPi3_MISO_SELECT_INPUT
	EPDC_D9 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA09 IOMUXC_ECSPi3_MISO_SELECT_INPUT
	SD2_DAT1 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA1 IOMUXC_ECSPi3_MISO_SELECT_INPUT
ECSPI3_MOSI	AUD_RXD (ALT1)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXD IOMUXC_ECSPi3_MOSI_SELECT_INPUT
	EPDC_D8 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA08 IOMUXC_ECSPi3_MOSI_SELECT_INPUT
	SD2_DAT0 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA0 IOMUXC_ECSPi3_MOSI_SELECT_INPUT
ECSPI3_RDY	AUD_MCLK (ALT2)	IOMUXC_SW_MUX_CTL_PAD_AUD_MCLK IOMUXC_ECSPi3_DATAREADY_B_SELECT_INPUT
	EPDC_D15 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA15 IOMUXC_ECSPi3_DATAREADY_B_SELECT_INPUT

*Table continues on the next page...*

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
ECSPI3_SCLK	AUD_TXD (ALT1)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXD IOMUXC_ECSPI3_CSPI_CLK_IN_SELECT_INPUT
	EPDC_D11 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA11 IOMUXC_ECSPI3_CSPI_CLK_IN_SELECT_INPUT
	SD2_CLK (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD2_CLK IOMUXC_ECSPI3_CSPI_CLK_IN_SELECT_INPUT
ECSPI3_SS0	AUD_RXFS (ALT6)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXFS IOMUXC_ECSPI3_SS0_SELECT_INPUT
	EPDC_D10 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA10 IOMUXC_ECSPI3_SS0_SELECT_INPUT
	SD2_CMD (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD2_CMD IOMUXC_ECSPI3_SS0_SELECT_INPUT
ECSPI3_SS1	AUD_RXC (ALT6)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXC IOMUXC_ECSPI3_SS1_SELECT_INPUT
	EPDC_D12 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA12 IOMUXC_ECSPI3_SS1_SELECT_INPUT
ECSPI3_SS2	EPDC_D13 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA13 IOMUXC_ECSPI3_SS2_SELECT_INPUT
	I2C1_SCL (ALT2)	IOMUXC_SW_MUX_CTL_PAD_I2C1_SCL IOMUXC_ECSPI3_SS2_SELECT_INPUT
ECSPI3_SS3	EPDC_D14 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA14 IOMUXC_ECSPI3_SS3_SELECT_INPUT
	I2C1_SDA (ALT2)	IOMUXC_SW_MUX_CTL_PAD_I2C1_SDA IOMUXC_ECSPI3_SS3_SELECT_INPUT
<b>ECSPI4 - Enhanced Configurable SPI</b>		
ECSPI4_MISO	EPDC_D1 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA01 IOMUXC_ECSPI4_MISO_SELECT_INPUT
	FEC_CRD_DV (ALT3)	IOMUXC_SW_MUX_CTL_PAD_FEC_CRD_DV IOMUXC_ECSPI4_MISO_SELECT_INPUT
	KEY_ROW1 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW1 IOMUXC_ECSPI4_MISO_SELECT_INPUT
ECSPI4_MOSI	EPDC_D0 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA00 IOMUXC_ECSPI4_MOSI_SELECT_INPUT
	FEC_RX_ER (ALT3)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_ER IOMUXC_ECSPI4_MOSI_SELECT_INPUT
	KEY_COL1 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL1 IOMUXC_ECSPI4_MOSI_SELECT_INPUT
ECSPI4_RDY	EPDC_D7 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA07

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
ECSPI4_SCLK	EPDC_D3 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA03 IOMUXC_ECSPi4_CSPI_CLK_IN_SELECT_INPUT
	FEC_TX_CLK (ALT3)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_CLK IOMUXC_ECSPi4_CSPI_CLK_IN_SELECT_INPUT
	KEY_ROW2 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW2 IOMUXC_ECSPi4_CSPI_CLK_IN_SELECT_INPUT
ECSPI4_SS0	EPDC_D2 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA02 IOMUXC_ECSPi4_SS0_SELECT_INPUT
	FEC_MDIO (ALT3)	IOMUXC_SW_MUX_CTL_PAD_FEC_MDIO IOMUXC_ECSPi4_SS0_SELECT_INPUT
	KEY_COL2 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL2 IOMUXC_ECSPi4_SS0_SELECT_INPUT
ECSPI4_SS1	EPDC_D4 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA04 IOMUXC_ECSPi4_SS1_SELECT_INPUT
	FEC_RXD1 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA1 IOMUXC_ECSPi4_SS1_SELECT_INPUT
ECSPI4_SS2	EPDC_D5 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA05 IOMUXC_ECSPi4_SS2_SELECT_INPUT
	FEC_TXD0 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA0 IOMUXC_ECSPi4_SS2_SELECT_INPUT
ECSPI4_SS3	EPDC_D6 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA06
<b>EIM - External Interface Module</b>		
EIM_ACLK_FREERUN	EPDC_PWRINT (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_IRQ
EIM_AD00	KEY_COL0 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL0
EIM_AD01	KEY_ROW0 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW0
EIM_AD02	KEY_COL1 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL1
EIM_AD03	KEY_ROW1 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW1
EIM_AD04	KEY_COL2 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL2
EIM_AD05	KEY_ROW2 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW2
EIM_AD06	KEY_COL3 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL3
EIM_AD07	KEY_ROW3 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW3
EIM_AD08	KEY_COL4 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL4
EIM_AD09	KEY_ROW4 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW4
EIM_AD10	KEY_COL5 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL5
EIM_AD11	KEY_ROW5 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW5
EIM_AD12	KEY_COL6 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL6
EIM_AD13	KEY_ROW6 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW6
EIM_AD14	KEY_COL7 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL7
EIM_AD15	KEY_ROW7 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW7

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
EIM_ADDR16	EPDC_D8 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA08
EIM_ADDR17	EPDC_D9 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA09
EIM_ADDR18	EPDC_D10 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA10
EIM_ADDR19	EPDC_D11 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA11
EIM_ADDR20	EPDC_D12 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA12
EIM_ADDR21	EPDC_D13 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA13
EIM_ADDR22	EPDC_D14 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA14
EIM_ADDR23	EPDC_D15 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA15
EIM_ADDR24	EPDC_VCOM0 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM0
EIM_ADDR25	EPDC_VCOM1 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM1
EIM_ADDR26	EPDC_BDR0 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR0
EIM_BCLK	EPDC_PWRCOM (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_COM
EIM_CRE	EPDC_BDR1 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR1
EIM_CS0_B	EPDC_PWRCTRL2 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL2
	LCD_HSYNC (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_HSYNC
EIM_CS1_B	EPDC_PWRCTRL3 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL3
	LCD_VSYNC (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_VSYNC
EIM_CS2_B	EPDC_SDCE0 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE0
EIM_CS3_B	LCD_DAT5 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA05
EIM_DATA00	LCD_DAT6 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA06
EIM_DATA01	LCD_DAT7 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA07
EIM_DATA02	LCD_DAT8 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA08
EIM_DATA03	LCD_DAT9 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA09
EIM_DATA04	LCD_DAT10 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA10
EIM_DATA05	LCD_DAT11 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA11
EIM_DATA06	LCD_DAT12 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA12
EIM_DATA07	LCD_DAT13 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA13
EIM_DATA08	LCD_DAT14 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA14
EIM_DATA09	LCD_DAT15 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA15
EIM_DATA10	LCD_DAT16 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA16
EIM_DATA11	LCD_DAT17 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA17
EIM_DATA12	LCD_DAT18 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA18
EIM_DATA13	LCD_DAT19 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA19
EIM_DATA14	LCD_DAT20 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA20
EIM_DATA15	LCD_DAT21 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA21
EIM_DTACK_B	EPDC_PWRWAKEUP (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_WAKE
		IOMUXC_EIM_DTACK_B_SELECT_INPUT
	LCD_RESET (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_RESET
		IOMUXC_EIM_DTACK_B_SELECT_INPUT

Table continues on the next page...



**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
EIM_EB0_B	EPDC_SDCE2 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE2
EIM_EB1_B	EPDC_SDCE3 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE3
EIM_EB2_B	LCD_DAT23 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA23
EIM_EB3_B	LCD_DAT22 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA22
EIM_LBA_B	EPDC_SDCE1 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE1
EIM_OE_B	EPDC_PWRCTRL1 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL1
	LCD_ENABLE (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_ENABLE
EIM_RW	EPDC_PWRCTRL0 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL0
	LCD_CLK (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_CLK
EIM_WAIT_B	EPDC_PWRSTAT (ALT3)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_STAT
		IOMUXC_EIM_WAIT_B_SELECT_INPUT
	LCD_RESET (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_RESET
		IOMUXC_EIM_WAIT_B_SELECT_INPUT
EPDC - Electrophoretic Display Controller		
EPDC_BDR0	ECSPI1_MISO (ALT3)	IOMUXC_SW_MUX_CTL_PAD_ECSP11_MISO
	EPDC_BDR0 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR0
EPDC_BDR1	ECSPI1_SS0 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_ECSP11_SS0
	EPDC_BDR1 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR1
EPDC_DATA00	EPDC_D0 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA00
EPDC_DATA01	EPDC_D1 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA01
EPDC_DATA02	EPDC_D2 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA02
EPDC_DATA03	EPDC_D3 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA03
EPDC_DATA04	EPDC_D4 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA04
EPDC_DATA05	EPDC_D5 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA05
EPDC_DATA06	EPDC_D6 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA06
EPDC_DATA07	EPDC_D7 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA07
EPDC_DATA08	EPDC_D8 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA08
EPDC_DATA09	EPDC_D9 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA09
EPDC_DATA10	EPDC_D10 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA10
EPDC_DATA11	EPDC_D11 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA11
EPDC_DATA12	EPDC_D12 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA12
EPDC_DATA13	EPDC_D13 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA13
EPDC_DATA14	EPDC_D14 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA14
EPDC_DATA15	EPDC_D15 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA15
EPDC_GDCLK	EPDC_GDCLK (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDCLK
EPDC_GDOE	EPDC_GDOE (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDOE
EPDC_GDRL	EPDC_GDRL (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDRL
EPDC_GDSP	EPDC_GDSP (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDSP

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
EPDC_PWR_COM	EPDC_D12 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA12
	EPDC_PWRCOM (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_COM
EPDC_PWR_CTRL0	EPDC_D8 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA08
	EPDC_PWRCTRL0 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL0
EPDC_PWR_CTRL1	EPDC_D9 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA09
	EPDC_PWRCTRL1 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL1
EPDC_PWR_CTRL2	EPDC_D10 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA10
	EPDC_PWRCTRL2 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL2
EPDC_PWR_CTRL3	EPDC_D11 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA11
	EPDC_PWRCTRL3 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL3
EPDC_PWR_IRQ	EPDC_D13 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA13
		IOMUXC_EPDC_EPDC_PWR_IRQ_SELECT_INPUT
	EPDC_PWRINT (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_IRQ IOMUXC_EPDC_EPDC_PWR_IRQ_SELECT_INPUT
EPDC_PWR_STAT	EPDC_D14 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA14
		IOMUXC_EPDC_EPDC_PWR_STAT_SELECT_INPUT
	EPDC_PWRSTAT (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_STAT IOMUXC_EPDC_EPDC_PWR_STAT_SELECT_INPUT
EPDC_PWR_WAKE	EPDC_D15 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA15
	EPDC_PWRWAKEUP (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_WAKE
EPDC_SDCE0	EPDC_SDCE0 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE0
EPDC_SDCE1	EPDC_SDCE1 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE1
EPDC_SDCE2	EPDC_SDCE2 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE2
EPDC_SDCE3	EPDC_SDCE3 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE3
EPDC_SDCE4	EPDC_SDSHR (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDSHR
	SD1_CLK (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD1_CLK
EPDC_SDCE5	EPDC_VCOM0 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM0
	SD1_CMD (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD1_CMD
EPDC_SDCE6	EPDC_VCOM1 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM1
	SD1_DAT0 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA0
EPDC_SDCE7	EPDC_BDR0 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR0
	SD1_DAT1 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA1
EPDC_SDCE8	EPDC_BDR1 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR1
	SD1_DAT2 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA2
EPDC_SDCE9	SD1_DAT3 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA3
EPDC_SDCLK_N	SD1_DAT4 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA4
EPDC_SDCLK_P	EPDC_SDCLK (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCLK
EPDC_SDLE	EPDC_SDLE (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDLE

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
EPDC_SDOE	EPDC_SDOE (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDOE
EPDC_SDOED	SD1_DAT5 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA5
EPDC_SDOEZ	SD1_DAT6 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA6
EPDC_SDSHR	EPDC_SDSHR (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDSHR
EPDC_VCOM0	ECSPI1_SCLK (ALT3)	IOMUXC_SW_MUX_CTL_PAD_ECSP11_SCLK
	EPDC_VCOM0 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM0
EPDC_VCOM1	ECSPI1_MOSI (ALT3)	IOMUXC_SW_MUX_CTL_PAD_ECSP11_MOSI
	EPDC_VCOM1 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM1
EPIT1 - Enhanced Periodic Interrupt Timer		
EPIT1_OUT	PWM1 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_PWM1
	SD2_CMD (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD2_CMD
	SD3_DAT2 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA2
EPIT2 - Enhanced Periodic Interrupt Timer		
EPIT2_OUT	LCD_DAT2 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA02
	SD3_DAT3 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA3
FEC - Fast Ethernet Controller		
FEC_COL	FEC_RXD1 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA1 IOMUXC_FEC_FEC_COL_SELECT_INPUT
	SD2_DAT2 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA2 IOMUXC_FEC_FEC_COL_SELECT_INPUT
	UART1_RXD (ALT3)	IOMUXC_SW_MUX_CTL_PAD_UART1_RXD IOMUXC_FEC_FEC_COL_SELECT_INPUT
FEC_MDC	AUD_MCLK (ALT3)	IOMUXC_SW_MUX_CTL_PAD_AUD_MCLK
	FEC_MDC (ALT0)	IOMUXC_SW_MUX_CTL_PAD_FEC_MDC
	SD1_DAT4 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA4
FEC_MDIO	AUD_RXFS (ALT3)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXFS IOMUXC_FEC_FEC_MDI_SELECT_INPUT
	FEC_MDIO (ALT0)	IOMUXC_SW_MUX_CTL_PAD_FEC_MDIO IOMUXC_FEC_FEC_MDI_SELECT_INPUT
	SD1_CLK (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD1_CLK IOMUXC_FEC_FEC_MDI_SELECT_INPUT
FEC_REF_OUT	FEC_REF_CLK (ALT0)	IOMUXC_SW_MUX_CTL_PAD_FEC_REF_CLK
	I2C2_SDA (ALT3)	IOMUXC_SW_MUX_CTL_PAD_I2C2_SDA
	PWM1 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_PWM1
	SD2_RST (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD2_RESET

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
FEC_RX_CLK	FEC_TXD1 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA1 IOMUXC_FEC_FEC_RX_CLK_SELECT_INPUT
	SD2_DAT3 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA3 IOMUXC_FEC_FEC_RX_CLK_SELECT_INPUT
	UART1_TXD (ALT3)	IOMUXC_SW_MUX_CTL_PAD_UART1_TXD IOMUXC_FEC_FEC_RX_CLK_SELECT_INPUT
FEC_RX_DATA0	FEC_RXD0 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA0 IOMUXC_FEC_FEC_RX_DATA0_SELECT_INPUT
	I2C1_SCL (ALT3)	IOMUXC_SW_MUX_CTL_PAD_I2C1_SCL IOMUXC_FEC_FEC_RX_DATA0_SELECT_INPUT
	SD1_DAT5 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA5 IOMUXC_FEC_FEC_RX_DATA0_SELECT_INPUT
FEC_RX_DATA1	AUD_TXFS (ALT3)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXFS IOMUXC_FEC_FEC_RX_DATA1_SELECT_INPUT
	FEC_RXD1 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA1 IOMUXC_FEC_FEC_RX_DATA1_SELECT_INPUT
	SD1_DAT2 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA2 IOMUXC_FEC_FEC_RX_DATA1_SELECT_INPUT
FEC_RX_DV	AUD_TXC (ALT3)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXC IOMUXC_FEC_FEC_RX_DV_SELECT_INPUT
	FEC_CRD_DV (ALT0)	IOMUXC_SW_MUX_CTL_PAD_FEC_CRD_DV IOMUXC_FEC_FEC_RX_DV_SELECT_INPUT
	SD1_DAT1 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA1 IOMUXC_FEC_FEC_RX_DV_SELECT_INPUT
FEC_RX_ER	AUD_RXD (ALT3)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXD IOMUXC_FEC_FEC_RX_ER_SELECT_INPUT
	FEC_RX_ER (ALT0)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_ER IOMUXC_FEC_FEC_RX_ER_SELECT_INPUT
	SD1_DAT0 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA0 IOMUXC_FEC_FEC_RX_ER_SELECT_INPUT
FEC_TX_CLK	AUD_RXC (ALT3)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXC IOMUXC_FEC_FEC_TX_CLK_SELECT_INPUT
	FEC_TX_CLK (ALT0)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_CLK IOMUXC_FEC_FEC_TX_CLK_SELECT_INPUT
	SD1_CMD (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD1_CMD IOMUXC_FEC_FEC_TX_CLK_SELECT_INPUT

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
FEC_TX_DATA0	AUD_TXD (ALT3)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXD
	FEC_TXD0 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA0
	SD1_DAT3 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA3
FEC_TX_DATA1	FEC_TXD1 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA1
	I2C2_SCL (ALT3)	IOMUXC_SW_MUX_CTL_PAD_I2C2_SCL
	SD1_DAT7 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA7
FEC_TX_EN	FEC_TX_EN (ALT0)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_EN
	I2C1_SDA (ALT3)	IOMUXC_SW_MUX_CTL_PAD_I2C1_SDA
	SD1_DAT6 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA6
<b>GPIO1 - General Purpose Input/Output</b>		
GPIO1_IO00	AUD_RXFS (ALT5)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXFS
GPIO1_IO01	AUD_RXC (ALT5)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXC
GPIO1_IO02	AUD_RXD (ALT5)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXD
GPIO1_IO03	AUD_TXC (ALT5)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXC
GPIO1_IO04	AUD_TXFS (ALT5)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXFS
GPIO1_IO05	AUD_TXD (ALT5)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXD
GPIO1_IO06	AUD_MCLK (ALT5)	IOMUXC_SW_MUX_CTL_PAD_AUD_MCLK
GPIO1_IO07	EPDC_D0 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA00
GPIO1_IO08	EPDC_D1 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA01
GPIO1_IO09	EPDC_D2 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA02
GPIO1_IO10	EPDC_D3 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA03
GPIO1_IO11	EPDC_D4 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA04
GPIO1_IO12	EPDC_D5 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA05
GPIO1_IO13	EPDC_D6 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA06
GPIO1_IO14	EPDC_D7 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA07
GPIO1_IO15	EPDC_D8 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA08
GPIO1_IO16	EPDC_D9 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA09
GPIO1_IO17	EPDC_D10 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA10
GPIO1_IO18	EPDC_D11 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA11
GPIO1_IO19	EPDC_D12 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA12
GPIO1_IO20	EPDC_D13 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA13
GPIO1_IO21	EPDC_D14 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA14
GPIO1_IO22	EPDC_D15 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA15
GPIO1_IO23	EPDC_SDCLK (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCLK
GPIO1_IO24	EPDC_SDLE (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDLE
GPIO1_IO25	EPDC_SDOE (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDOE
GPIO1_IO26	EPDC_SDSHR (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDSHR
GPIO1_IO27	EPDC_SDCE0 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE0
GPIO1_IO28	EPDC_SDCE1 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE1

*Table continues on the next page...*

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
GPIO1_IO29	EPDC_SDCE2 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE2
GPIO1_IO30	EPDC_SDCE3 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE3
GPIO1_IO31	EPDC_GDCLK (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDCLK
GPIO2 - General Purpose Input/Output		
GPIO2_IO00	EPDC_GDOE (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDOE
GPIO2_IO01	EPDC_GDRL (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDRL
GPIO2_IO02	EPDC_GDSP (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDSP
GPIO2_IO03	EPDC_VCOM0 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM0
GPIO2_IO04	EPDC_VCOM1 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM1
GPIO2_IO05	EPDC_BDR0 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR0
GPIO2_IO06	EPDC_BDR1 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR1
GPIO2_IO07	EPDC_PWRCTRL0 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL0
GPIO2_IO08	EPDC_PWRCTRL1 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL1
GPIO2_IO09	EPDC_PWRCTRL2 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL2
GPIO2_IO10	EPDC_PWRCTRL3 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL3
GPIO2_IO11	EPDC_PWRCOM (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_COM
GPIO2_IO12	EPDC_PWRINT (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_IRQ
GPIO2_IO13	EPDC_PWRSTAT (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_STAT
GPIO2_IO14	EPDC_PWRWAKEUP (ALT5)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_WAKE
GPIO2_IO15	LCD_CLK (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_CLK
GPIO2_IO16	LCD_ENABLE (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_ENABLE
GPIO2_IO17	LCD_HSYNC (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_HSYNC
GPIO2_IO18	LCD_VSYNC (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_VSYNC
GPIO2_IO19	LCD_RESET (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_RESET
GPIO2_IO20	LCD_DAT0 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA00
GPIO2_IO21	LCD_DAT1 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA01
GPIO2_IO22	LCD_DAT2 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA02
GPIO2_IO23	LCD_DAT3 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA03
GPIO2_IO24	LCD_DAT4 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA04
GPIO2_IO25	LCD_DAT5 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA05
GPIO2_IO26	LCD_DAT6 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA06
GPIO2_IO27	LCD_DAT7 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA07
GPIO2_IO28	LCD_DAT8 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA08
GPIO2_IO29	LCD_DAT9 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA09
GPIO2_IO30	LCD_DAT10 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA10
GPIO2_IO31	LCD_DAT11 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA11
GPIO3 - General Purpose Input/Output		
GPIO3_IO00	LCD_DAT12 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA12
GPIO3_IO01	LCD_DAT13 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA13

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
GPIO3_IO02	LCD_DAT14 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA14
GPIO3_IO03	LCD_DAT15 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA15
GPIO3_IO04	LCD_DAT16 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA16
GPIO3_IO05	LCD_DAT17 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA17
GPIO3_IO06	LCD_DAT18 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA18
GPIO3_IO07	LCD_DAT19 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA19
GPIO3_IO08	LCD_DAT20 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA20
GPIO3_IO09	LCD_DAT21 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA21
GPIO3_IO10	LCD_DAT22 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA22
GPIO3_IO11	LCD_DAT23 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA23
GPIO3_IO12	I2C1_SCL (ALT5)	IOMUXC_SW_MUX_CTL_PAD_I2C1_SCL
GPIO3_IO13	I2C1_SDA (ALT5)	IOMUXC_SW_MUX_CTL_PAD_I2C1_SDA
GPIO3_IO14	I2C2_SCL (ALT5)	IOMUXC_SW_MUX_CTL_PAD_I2C2_SCL
GPIO3_IO15	I2C2_SDA (ALT5)	IOMUXC_SW_MUX_CTL_PAD_I2C2_SDA
GPIO3_IO16	UART1_RXD (ALT5)	IOMUXC_SW_MUX_CTL_PAD_UART1_RXD
GPIO3_IO17	UART1_TXD (ALT5)	IOMUXC_SW_MUX_CTL_PAD_UART1_TXD
GPIO3_IO18	WDOG_B (ALT5)	IOMUXC_SW_MUX_CTL_PAD_WDOG_B
GPIO3_IO19	HSIC_DAT (ALT5)	IOMUXC_SW_MUX_CTL_PAD_USB_H_DATA
GPIO3_IO20	HSIC_STROBE (ALT5)	IOMUXC_SW_MUX_CTL_PAD_USB_H_STROBE
GPIO3_IO21	REF_CLK_24M (ALT5)	IOMUXC_SW_MUX_CTL_PAD_REF_CLK_24M
GPIO3_IO22	REF_CLK_32K (ALT5)	IOMUXC_SW_MUX_CTL_PAD_REF_CLK_32K
GPIO3_IO23	PWM1 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_PWM1
GPIO3_IO24	KEY_COL0 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL0
GPIO3_IO25	KEY_ROW0 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW0
GPIO3_IO26	KEY_COL1 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL1
GPIO3_IO27	KEY_ROW1 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW1
GPIO3_IO28	KEY_COL2 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL2
GPIO3_IO29	KEY_ROW2 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW2
GPIO3_IO30	KEY_COL3 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL3
GPIO3_IO31	KEY_ROW3 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW3
<b>GPIO4 - General Purpose Input/Output</b>		
GPIO4_IO00	KEY_COL4 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL4
GPIO4_IO01	KEY_ROW4 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW4
GPIO4_IO02	KEY_COL5 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL5
GPIO4_IO03	KEY_ROW5 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW5
GPIO4_IO04	KEY_COL6 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL6
GPIO4_IO05	KEY_ROW6 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW6
GPIO4_IO06	KEY_COL7 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL7
GPIO4_IO07	KEY_ROW7 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW7

*Table continues on the next page...*

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
GPIO4_IO08	ECSPI1_SCLK (ALT5)	IOMUXC_SW_MUX_CTL_PAD_ECSP11_SCLK
GPIO4_IO09	ECSPI1_MOSI (ALT5)	IOMUXC_SW_MUX_CTL_PAD_ECSP11_MOSI
GPIO4_IO10	ECSPI1_MISO (ALT5)	IOMUXC_SW_MUX_CTL_PAD_ECSP11_MISO
GPIO4_IO11	ECSPI1_SS0 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_ECSP11_SS0
GPIO4_IO12	ECSPI2_SCLK (ALT5)	IOMUXC_SW_MUX_CTL_PAD_ECSP12_SCLK
GPIO4_IO13	ECSPI2_MOSI (ALT5)	IOMUXC_SW_MUX_CTL_PAD_ECSP12_MOSI
GPIO4_IO14	ECSPI2_MISO (ALT5)	IOMUXC_SW_MUX_CTL_PAD_ECSP12_MISO
GPIO4_IO15	ECSPI2_SS0 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_ECSP12_SS0
GPIO4_IO16	FEC_TXD1 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA1
GPIO4_IO17	FEC_RXD0 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA0
GPIO4_IO18	FEC_RXD1 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA1
GPIO4_IO19	FEC_RX_ER (ALT5)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_ER
GPIO4_IO20	FEC_MDIO (ALT5)	IOMUXC_SW_MUX_CTL_PAD_FEC_MDIO
GPIO4_IO21	FEC_TX_CLK (ALT5)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_CLK
GPIO4_IO22	FEC_TX_EN (ALT5)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_EN
GPIO4_IO23	FEC_MDC (ALT5)	IOMUXC_SW_MUX_CTL_PAD_FEC_MDC
GPIO4_IO24	FEC_TXD0 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA0
GPIO4_IO25	FEC CRS_DV (ALT5)	IOMUXC_SW_MUX_CTL_PAD_FEC_CRD_DV
GPIO4_IO26	FEC_REF_CLK (ALT5)	IOMUXC_SW_MUX_CTL_PAD_FEC_REF_CLK
GPIO4_IO27	SD2_RST (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD2_RESET
GPIO4_IO28	SD2_DAT3 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA3
GPIO4_IO29	SD2_DAT6 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA6
GPIO4_IO30	SD2_DAT1 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA1
GPIO4_IO31	SD2_DAT5 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA5
<b>GPIO5 - General Purpose Input/Output</b>		
GPIO5_IO00	SD2_DAT7 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA7
GPIO5_IO01	SD2_DAT0 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA0
GPIO5_IO02	SD2_DAT4 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA4
GPIO5_IO03	SD2_DAT2 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA2
GPIO5_IO04	SD2_CMD (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD2_CMD
GPIO5_IO05	SD2_CLK (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD2_CLK
GPIO5_IO06	SD1_DAT3 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA3
GPIO5_IO07	SD1_DAT6 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA6
GPIO5_IO08	SD1_DAT1 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA1
GPIO5_IO09	SD1_DAT5 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA5
GPIO5_IO10	SD1_DAT7 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA7
GPIO5_IO11	SD1_DAT0 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA0
GPIO5_IO12	SD1_DAT4 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA4
GPIO5_IO13	SD1_DAT2 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA2

Table continues on the next page...



**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
GPIO5_IO14	SD1_CMD (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD1_CMD
GPIO5_IO15	SD1_CLK (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD1_CLK
GPIO5_IO16	SD3_DAT2 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA2
GPIO5_IO17	SD3_DAT3 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA3
GPIO5_IO18	SD3_CLK (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD3_CLK
GPIO5_IO19	SD3_DAT0 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA0
GPIO5_IO20	SD3_DAT1 (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA1
GPIO5_IO21	SD3_CMD (ALT5)	IOMUXC_SW_MUX_CTL_PAD_SD3_CMD
<b>GPT - General Purpose Timer</b>		
GPT_CAPTURE1	FEC_MDIO (ALT4)	IOMUXC_SW_MUX_CTL_PAD_FEC_MDIO IOMUXC_GPT_CAPIN1_SELECT_INPUT
	LCD_DAT18 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA18 IOMUXC_GPT_CAPIN1_SELECT_INPUT
GPT_CAPTURE2	FEC_TX_CLK (ALT4)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_CLK IOMUXC_GPT_CAPIN2_SELECT_INPUT
	LCD_DAT19 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA19 IOMUXC_GPT_CAPIN2_SELECT_INPUT
GPT_CLKIN	FEC_TXD0 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA0 IOMUXC_GPT_CLKIN_SELECT_INPUT
	LCD_DAT23 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA23 IOMUXC_GPT_CLKIN_SELECT_INPUT
GPT_COMPARE1	FEC_RX_ER (ALT4)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_ER
	LCD_DAT20 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA20
GPT_COMPARE2	FEC CRS_DV (ALT4)	IOMUXC_SW_MUX_CTL_PAD_FEC_CRD_DV
	LCD_DAT21 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA21
GPT_COMPARE3	FEC_RXD1 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA1
	LCD_DAT22 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA22
<b>I2C1 - I2C Controller</b>		
I2C1_SCL	AUD_RXFS (ALT1)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXFS IOMUXC_I2C1_SCL_IN_SELECT_INPUT
	I2C1_SCL (ALT0)	IOMUXC_SW_MUX_CTL_PAD_I2C1_SCL IOMUXC_I2C1_SCL_IN_SELECT_INPUT
	HSIC_DAT (ALT1)	IOMUXC_SW_MUX_CTL_PAD_USB_H_DATA IOMUXC_I2C1_SCL_IN_SELECT_INPUT

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
I2C1_SDA	AUD_RXC (ALT1)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXC IOMUXC_I2C1_SDA_IN_SELECT_INPUT
	I2C1_SDA (ALT0)	IOMUXC_SW_MUX_CTL_PAD_I2C1_SDA IOMUXC_I2C1_SDA_IN_SELECT_INPUT
	HSIC_STROBE (ALT1)	IOMUXC_SW_MUX_CTL_PAD_USB_H_STROBE IOMUXC_I2C1_SDA_IN_SELECT_INPUT
I2C2 - I2C Controller		
I2C2_SCL	EPDC_SDCLK (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCLK IOMUXC_I2C2_SCL_IN_SELECT_INPUT
	I2C2_SCL (ALT0)	IOMUXC_SW_MUX_CTL_PAD_I2C2_SCL IOMUXC_I2C2_SCL_IN_SELECT_INPUT
	KEY_COL0 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL0 IOMUXC_I2C2_SCL_IN_SELECT_INPUT
	LCD_DAT16 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA16 IOMUXC_I2C2_SCL_IN_SELECT_INPUT
I2C2_SDA	EPDC_SDLE (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDLE IOMUXC_I2C2_SDA_IN_SELECT_INPUT
	I2C2_SDA (ALT0)	IOMUXC_SW_MUX_CTL_PAD_I2C2_SDA IOMUXC_I2C2_SDA_IN_SELECT_INPUT
	KEY_ROW0 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW0 IOMUXC_I2C2_SDA_IN_SELECT_INPUT
	LCD_DAT17 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA17 IOMUXC_I2C2_SDA_IN_SELECT_INPUT
I2C3 - I2C Controller		
I2C3_SCL	AUD_RXFS (ALT4)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXFS IOMUXC_I2C3_SCL_IN_SELECT_INPUT
	EPDC_SDCE2 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE2 IOMUXC_I2C3_SCL_IN_SELECT_INPUT
	REF_CLK_24M (ALT1)	IOMUXC_SW_MUX_CTL_PAD_REF_CLK_24M IOMUXC_I2C3_SCL_IN_SELECT_INPUT
I2C3_SDA	AUD_RXC (ALT4)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXC IOMUXC_I2C3_SDA_IN_SELECT_INPUT
	EPDC_SDCE3 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE3 IOMUXC_I2C3_SDA_IN_SELECT_INPUT
	REF_CLK_32K (ALT1)	IOMUXC_SW_MUX_CTL_PAD_REF_CLK_32K IOMUXC_I2C3_SDA_IN_SELECT_INPUT
KPP - Keypad Port		

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
KEY_COL0	KEY_COL0 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL0 IOMUXC_KEY_COL0_SELECT_INPUT
	LCD_DAT8 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA08 IOMUXC_KEY_COL0_SELECT_INPUT
	SD1_CLK (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD1_CLK IOMUXC_KEY_COL0_SELECT_INPUT
KEY_COL1	KEY_COL1 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL1 IOMUXC_KEY_COL1_SELECT_INPUT
	LCD_DAT10 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA10 IOMUXC_KEY_COL1_SELECT_INPUT
	SD1_DAT0 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA0 IOMUXC_KEY_COL1_SELECT_INPUT
KEY_COL2	KEY_COL2 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL2 IOMUXC_KEY_COL2_SELECT_INPUT
	LCD_DAT12 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA12 IOMUXC_KEY_COL2_SELECT_INPUT
	SD1_DAT2 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA2 IOMUXC_KEY_COL2_SELECT_INPUT
KEY_COL3	KEY_COL3 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL3 IOMUXC_KEY_COL3_SELECT_INPUT
	LCD_DAT14 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA14 IOMUXC_KEY_COL3_SELECT_INPUT
	SD1_DAT4 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA4 IOMUXC_KEY_COL3_SELECT_INPUT
KEY_COL4	KEY_COL4 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL4 IOMUXC_KEY_COL4_SELECT_INPUT
	LCD_DAT16 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA16 IOMUXC_KEY_COL4_SELECT_INPUT
	SD1_DAT6 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA6 IOMUXC_KEY_COL4_SELECT_INPUT
KEY_COL5	KEY_COL5 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL5 IOMUXC_KEY_COL5_SELECT_INPUT
	LCD_DAT18 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA18 IOMUXC_KEY_COL5_SELECT_INPUT
	SD3_CLK (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD3_CLK IOMUXC_KEY_COL5_SELECT_INPUT

*Table continues on the next page...*

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
KEY_COL6	KEY_COL6 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL6 IOMUXC_KEY_COL6_SELECT_INPUT
	LCD_DAT20 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA20 IOMUXC_KEY_COL6_SELECT_INPUT
	SD3_DAT0 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA0 IOMUXC_KEY_COL6_SELECT_INPUT
KEY_COL7	KEY_COL7 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL7 IOMUXC_KEY_COL7_SELECT_INPUT
	LCD_DAT22 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA22 IOMUXC_KEY_COL7_SELECT_INPUT
	SD3_DAT2 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA2 IOMUXC_KEY_COL7_SELECT_INPUT
KEY_ROW0	KEY_ROW0 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW0 IOMUXC_KEY_ROW0_SELECT_INPUT
	LCD_DAT9 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA09 IOMUXC_KEY_ROW0_SELECT_INPUT
	SD1_CMD (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD1_CMD IOMUXC_KEY_ROW0_SELECT_INPUT
KEY_ROW1	KEY_ROW1 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW1 IOMUXC_KEY_ROW1_SELECT_INPUT
	LCD_DAT11 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA11 IOMUXC_KEY_ROW1_SELECT_INPUT
	SD1_DAT1 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA1 IOMUXC_KEY_ROW1_SELECT_INPUT
KEY_ROW2	KEY_ROW2 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW2 IOMUXC_KEY_ROW2_SELECT_INPUT
	LCD_DAT13 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA13 IOMUXC_KEY_ROW2_SELECT_INPUT
	SD1_DAT3 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA3 IOMUXC_KEY_ROW2_SELECT_INPUT
KEY_ROW3	KEY_ROW3 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW3 IOMUXC_KEY_ROW3_SELECT_INPUT
	LCD_DAT15 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA15 IOMUXC_KEY_ROW3_SELECT_INPUT
	SD1_DAT5 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA5 IOMUXC_KEY_ROW3_SELECT_INPUT

*Table continues on the next page...*

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
KEY_ROW4	KEY_ROW4 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW4 IOMUXC_KEY_ROW4_SELECT_INPUT
	LCD_DAT17 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA17 IOMUXC_KEY_ROW4_SELECT_INPUT
	SD1_DAT7 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA7 IOMUXC_KEY_ROW4_SELECT_INPUT
KEY_ROW5	KEY_ROW5 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW5 IOMUXC_KEY_ROW5_SELECT_INPUT
	LCD_DAT19 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA19 IOMUXC_KEY_ROW5_SELECT_INPUT
	SD3_CMD (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD3_CMD IOMUXC_KEY_ROW5_SELECT_INPUT
KEY_ROW6	KEY_ROW6 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW6 IOMUXC_KEY_ROW6_SELECT_INPUT
	LCD_DAT21 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA21 IOMUXC_KEY_ROW6_SELECT_INPUT
	SD3_DAT1 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA1 IOMUXC_KEY_ROW6_SELECT_INPUT
KEY_ROW7	KEY_ROW7 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW7 IOMUXC_KEY_ROW7_SELECT_INPUT
	LCD_DAT23 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA23 IOMUXC_KEY_ROW7_SELECT_INPUT
	SD3_DAT3 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA3 IOMUXC_KEY_ROW7_SELECT_INPUT
LCD - LCD Interface		
LCD_BUSY	LCD_RESET (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_RESET IOMUXC_LCD_BUSY_SELECT_INPUT
LCD_CLK	LCD_CLK (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_CLK
LCD_CS	LCD_HSYNC (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_HSYNC
LCD_DATA00	KEY_COL0 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL0 IOMUXC_LCD_DATA00_SELECT_INPUT
	LCD_DAT0 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA00 IOMUXC_LCD_DATA00_SELECT_INPUT
LCD_DATA01	KEY_ROW0 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW0 IOMUXC_LCD_DATA01_SELECT_INPUT
	LCD_DAT1 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA01 IOMUXC_LCD_DATA01_SELECT_INPUT

*Table continues on the next page...*

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
LCD_DATA02	KEY_COL1 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL1 IOMUXC_LCD_DATA02_SELECT_INPUT
	LCD_DAT2 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA02 IOMUXC_LCD_DATA02_SELECT_INPUT
LCD_DATA03	KEY_ROW1 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW1 IOMUXC_LCD_DATA03_SELECT_INPUT
	LCD_DAT3 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA03 IOMUXC_LCD_DATA03_SELECT_INPUT
LCD_DATA04	KEY_COL2 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL2 IOMUXC_LCD_DATA04_SELECT_INPUT
	LCD_DAT4 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA04 IOMUXC_LCD_DATA04_SELECT_INPUT
LCD_DATA05	KEY_ROW2 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW2 IOMUXC_LCD_DATA05_SELECT_INPUT
	LCD_DAT5 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA05 IOMUXC_LCD_DATA05_SELECT_INPUT
LCD_DATA06	KEY_COL3 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL3 IOMUXC_LCD_DATA06_SELECT_INPUT
	LCD_DAT6 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA06 IOMUXC_LCD_DATA06_SELECT_INPUT
LCD_DATA07	KEY_ROW3 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW3 IOMUXC_LCD_DATA07_SELECT_INPUT
	LCD_DAT7 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA07 IOMUXC_LCD_DATA07_SELECT_INPUT
LCD_DATA08	KEY_COL4 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL4 IOMUXC_LCD_DATA08_SELECT_INPUT
	LCD_DAT8 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA08 IOMUXC_LCD_DATA08_SELECT_INPUT
LCD_DATA09	KEY_ROW4 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW4 IOMUXC_LCD_DATA09_SELECT_INPUT
	LCD_DAT9 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA09 IOMUXC_LCD_DATA09_SELECT_INPUT
LCD_DATA10	KEY_COL5 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL5 IOMUXC_LCD_DATA10_SELECT_INPUT
	LCD_DAT10 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA10 IOMUXC_LCD_DATA10_SELECT_INPUT

*Table continues on the next page...*

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
LCD_DATA11	KEY_ROW5 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW5 IOMUXC_LCD_DATA11_SELECT_INPUT
	LCD_DAT11 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA11 IOMUXC_LCD_DATA11_SELECT_INPUT
LCD_DATA12	KEY_COL6 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL6 IOMUXC_LCD_DATA12_SELECT_INPUT
	LCD_DAT12 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA12 IOMUXC_LCD_DATA12_SELECT_INPUT
LCD_DATA13	KEY_ROW6 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW6 IOMUXC_LCD_DATA13_SELECT_INPUT
	LCD_DAT13 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA13 IOMUXC_LCD_DATA13_SELECT_INPUT
LCD_DATA14	KEY_COL7 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL7 IOMUXC_LCD_DATA14_SELECT_INPUT
	LCD_DAT14 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA14 IOMUXC_LCD_DATA14_SELECT_INPUT
LCD_DATA15	KEY_ROW7 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW7 IOMUXC_LCD_DATA15_SELECT_INPUT
	LCD_DAT15 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA15 IOMUXC_LCD_DATA15_SELECT_INPUT
LCD_DATA16	EPDC_PWRCTRL0 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL0 IOMUXC_LCD_DATA16_SELECT_INPUT
	LCD_DAT16 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA16 IOMUXC_LCD_DATA16_SELECT_INPUT
LCD_DATA17	EPDC_PWRCTRL1 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL1 IOMUXC_LCD_DATA17_SELECT_INPUT
	LCD_DAT17 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA17 IOMUXC_LCD_DATA17_SELECT_INPUT
LCD_DATA18	EPDC_PWRCTRL2 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL2 IOMUXC_LCD_DATA18_SELECT_INPUT
	LCD_DAT18 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA18 IOMUXC_LCD_DATA18_SELECT_INPUT
LCD_DATA19	EPDC_PWRCTRL3 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL3 IOMUXC_LCD_DATA19_SELECT_INPUT
	LCD_DAT19 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA19 IOMUXC_LCD_DATA19_SELECT_INPUT

*Table continues on the next page...*

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
LCD_DATA20	EPDC_PWRCOM (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_COM IOMUXC_LCD_DATA20_SELECT_INPUT
	LCD_DAT20 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA20 IOMUXC_LCD_DATA20_SELECT_INPUT
LCD_DATA21	EPDC_PWRINT (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_IRQ IOMUXC_LCD_DATA21_SELECT_INPUT
	LCD_DAT21 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA21 IOMUXC_LCD_DATA21_SELECT_INPUT
LCD_DATA22	EPDC_PWRSTAT (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_STAT IOMUXC_LCD_DATA22_SELECT_INPUT
	LCD_DAT22 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA22 IOMUXC_LCD_DATA22_SELECT_INPUT
LCD_DATA23	EPDC_PWRWAKEUP (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_WAKE IOMUXC_LCD_DATA23_SELECT_INPUT
	LCD_DAT23 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA23 IOMUXC_LCD_DATA23_SELECT_INPUT
LCD_DATA24	EPDC_D0 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA00
LCD_DATA25	EPDC_D1 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA01
LCD_DATA26	EPDC_D2 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA02
LCD_DATA27	EPDC_D3 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA03
LCD_DATA28	EPDC_D4 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA04
LCD_DATA29	EPDC_D5 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA05
LCD_DATA30	EPDC_D6 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA06
LCD_DATA31	EPDC_D7 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA07
LCD_ENABLE	LCD_ENABLE (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_ENABLE
LCD_HSYNC	LCD_HSYNC (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_HSYNC IOMUXC_LCD_BUSY_SELECT_INPUT
LCD_RD_E	LCD_ENABLE (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_ENABLE
LCD_RESET	LCD_RESET (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_RESET
LCD_RS	LCD_VSYNC (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_VSYNC
LCD_VSYNC	LCD_VSYNC (ALT0)	IOMUXC_SW_MUX_CTL_PAD_LCD_VSYNC
LCD_WR_RWN	LCD_CLK (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_CLK
<b>MMDC - Multi Mode DDR Controller</b>		
DRAM_ADDR00	DRAM_A0	Not multiplexed.
DRAM_ADDR01	DRAM_A1	Not multiplexed.
DRAM_ADDR02	DRAM_A2	Not multiplexed.
DRAM_ADDR03	DRAM_A3	Not multiplexed.
DRAM_ADDR04	DRAM_A4	Not multiplexed.
DRAM_ADDR05	DRAM_A5	Not multiplexed.

*Table continues on the next page...*



**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
DRAM_ADDR06	DRAM_A6	Not multiplexed.
DRAM_ADDR07	DRAM_A7	Not multiplexed.
DRAM_ADDR08	DRAM_A8	Not multiplexed.
DRAM_ADDR09	DRAM_A9	Not multiplexed.
DRAM_ADDR10	DRAM_A10	Not multiplexed.
DRAM_ADDR11	DRAM_A11	Not multiplexed.
DRAM_ADDR12	DRAM_A12	Not multiplexed.
DRAM_ADDR13	DRAM_A13	Not multiplexed.
DRAM_ADDR14	DRAM_A14	Not multiplexed.
DRAM_ADDR15	DRAM_A15	Not multiplexed.
DRAM_CAS_B	DRAM_CAS	Not multiplexed.
DRAM_CS0_B	DRAM_CS0	Not multiplexed.
DRAM_CS1_B	DRAM_CS1	Not multiplexed.
DRAM_DATA00	DRAM_D0	Not multiplexed.
DRAM_DATA01	DRAM_D1	Not multiplexed.
DRAM_DATA02	DRAM_D2	Not multiplexed.
DRAM_DATA03	DRAM_D3	Not multiplexed.
DRAM_DATA04	DRAM_D4	Not multiplexed.
DRAM_DATA05	DRAM_D5	Not multiplexed.
DRAM_DATA06	DRAM_D6	Not multiplexed.
DRAM_DATA07	DRAM_D7	Not multiplexed.
DRAM_DATA08	DRAM_D8	Not multiplexed.
DRAM_DATA09	DRAM_D9	Not multiplexed.
DRAM_DATA10	DRAM_D10	Not multiplexed.
DRAM_DATA11	DRAM_D11	Not multiplexed.
DRAM_DATA12	DRAM_D12	Not multiplexed.
DRAM_DATA13	DRAM_D13	Not multiplexed.
DRAM_DATA14	DRAM_D14	Not multiplexed.
DRAM_DATA15	DRAM_D15	Not multiplexed.
DRAM_DATA16	DRAM_D16	Not multiplexed.
DRAM_DATA17	DRAM_D17	Not multiplexed.
DRAM_DATA18	DRAM_D18	Not multiplexed.
DRAM_DATA19	DRAM_D19	Not multiplexed.
DRAM_DATA20	DRAM_D20	Not multiplexed.
DRAM_DATA21	DRAM_D21	Not multiplexed.
DRAM_DATA22	DRAM_D22	Not multiplexed.
DRAM_DATA23	DRAM_D23	Not multiplexed.
DRAM_DATA24	DRAM_D24	Not multiplexed.
DRAM_DATA25	DRAM_D25	Not multiplexed.

*Table continues on the next page...*

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
DRAM_DATA26	DRAM_D26	Not multiplexed.
DRAM_DATA27	DRAM_D27	Not multiplexed.
DRAM_DATA28	DRAM_D28	Not multiplexed.
DRAM_DATA29	DRAM_D29	Not multiplexed.
DRAM_DATA30	DRAM_D30	Not multiplexed.
DRAM_DATA31	DRAM_D31	Not multiplexed.
DRAM_DQM0	DRAM_DQM0	Not multiplexed.
DRAM_DQM1	DRAM_DQM1	Not multiplexed.
DRAM_DQM2	DRAM_DQM2	Not multiplexed.
DRAM_DQM3	DRAM_DQM3	Not multiplexed.
DRAM_ODT0	DRAM_SDODT0	Not multiplexed.
DRAM_ODT1	DRAM_SDODT1	Not multiplexed.
DRAM_RAS_B	DRAM_RAS	Not multiplexed.
DRAM_RESET	DRAM_RESET	Not multiplexed.
DRAM_SDBA0	DRAM_SDBA0	Not multiplexed.
DRAM_SDBA1	DRAM_SDBA1	Not multiplexed.
DRAM_SDBA2	DRAM_SDBA2	Not multiplexed.
DRAM_SDCKE0	DRAM_SDCKE0	Not multiplexed.
DRAM_SDCKE1	DRAM_SDCKE1	Not multiplexed.
DRAM_SDCLK0_N	DRAM_SDCLK_0_B	Not multiplexed.
DRAM_SDCLK0_P	DRAM_SDCLK_0	Not multiplexed.
DRAM_SDQS0_N	DRAM_SDQS0_B	Not multiplexed.
DRAM_SDQS0_P	DRAM_SDQS0	Not multiplexed.
DRAM_SDQS1_N	DRAM_SDQS1_B	Not multiplexed.
DRAM_SDQS1_P	DRAM_SDQS1	Not multiplexed.
DRAM_SDQS2_N	DRAM_SDQS2_B	Not multiplexed.
DRAM_SDQS2_P	DRAM_SDQS2	Not multiplexed.
DRAM_SDQS3_N	DRAM_SDQS3_B	Not multiplexed.
DRAM_SDQS3_P	DRAM_SDQS3	Not multiplexed.
DRAM_SDWE_B	DRAM_SDWE	Not multiplexed.
PMU - Power Management Unit		
DRAM_VREF	DDR_VREF	Not multiplexed.
DRAM_ZQPAD	ZQPAD	Not multiplexed.
GND	GND	Not multiplexed.
GPANAIO	GPANAIO	Not multiplexed.
NC	NC	Not multiplexed.
NVCC_1P2	NVCC_1P2	Not multiplexed.
NVCC_DRAM	NVCC_DRAM	Not multiplexed.
NVCC_DRAM_2P5	NVCC_DRAM2P5	Not multiplexed.

*Table continues on the next page...*

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
NVCC_IO_1P8	NVCC18_IO	Not multiplexed.
NVCC_IO_3P3	NVCC33_IO	Not multiplexed.
NVCC_PLL	NVCC_PLL_OUT	Not multiplexed.
USB_OTG1_VBUS	USB_OTG1_VBUS	Not multiplexed.
USB_OTG2_VBUS	USB_OTG2_VBUS	Not multiplexed.
VDD_ARM_CAP	VDD_ARM_CAP	Not multiplexed.
VDD_ARM_IN	VDD_ARM_IN	Not multiplexed.
VDD_HIGH_CAP	VDDHIGH_CAP	Not multiplexed.
VDD_HIGH_IN	VDDHIGH_IN	Not multiplexed.
VDD_PU_CAP	VDD_PU_CAP	Not multiplexed.
VDD_PU_IN	VDD_PU_IN	Not multiplexed.
VDD_SNV5_CAP	VDD_SNV5_CAP	Not multiplexed.
VDD_SNV5_IN	VDD_SNV5_IN	Not multiplexed.
VDD_SOC_CAP	VDD_SOC_CAP	Not multiplexed.
VDD_SOC_IN	VDD_SOC_IN	Not multiplexed.
VDD_USB_CAP	VDD_USB_CAP	Not multiplexed.
PWM1 - Pulse Width Modulation		
PWM1_OUT	EPDC_SDCE2 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE2
	LCD_DAT0 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA00
	PWM1 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_PWM1
	UART1_RXD (ALT1)	IOMUXC_SW_MUX_CTL_PAD_UART1_RXD
	HSIC_DAT (ALT2)	IOMUXC_SW_MUX_CTL_PAD_USB_H_DATA
PWM2 - Pulse Width Modulation		
PWM2_OUT	EPDC_SDCE3 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE3
	LCD_DAT1 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA01
	UART1_TXD (ALT1)	IOMUXC_SW_MUX_CTL_PAD_UART1_TXD
	HSIC_STROBE (ALT2)	IOMUXC_SW_MUX_CTL_PAD_USB_H_STROBE
PWM3 - Pulse Width Modulation		
PWM3_OUT	AUD_TXFS (ALT1)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXFS
	EPDC_SDCE0 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE0
	LCD_DAT2 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA02
	REF_CLK_24M (ALT2)	IOMUXC_SW_MUX_CTL_PAD_REF_CLK_24M
PWM4 - Pulse Width Modulation		

*Table continues on the next page...*

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
PWM4_OUT	AUD_MCLK (ALT1)	IOMUXC_SW_MUX_CTL_PAD_AUD_MCLK
	EPDC_GDSP (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDSP
	EPDC_SDCE1 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE1
	FEC_REF_CLK (ALT3)	IOMUXC_SW_MUX_CTL_PAD_FEC_REF_CLK
	LCD_CLK (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_CLK
	LCD_DAT3 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA03
	REF_CLK_32K (ALT2)	IOMUXC_SW_MUX_CTL_PAD_REF_CLK_32K
SDMA - Smart Direct Memory Access Controller		
SDMA_EXT_EVENT0	ECSPI2_MISO (ALT1)	IOMUXC_SW_MUX_CTL_PAD_ECSPi2_MISO
SDMA_EXT_EVENT1	ECSPI2_MOSI (ALT1)	IOMUXC_SW_MUX_CTL_PAD_ECSPi2_MOSI
SJC - System JTAG Controller		
JTAG_DE_B	SD3_DAT1 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA1
JTAG_MOD	JTAG_MOD	Not multiplexed.
JTAG_TCK	JTAG_TCK	Not multiplexed.
JTAG_TDI	JTAG_TDI	Not multiplexed.
JTAG_TDO	JTAG_TDO	Not multiplexed.
JTAG_TMS	JTAG_TMS	Not multiplexed.
JTAG_TRSTB	JTAG_TRSTB	Not multiplexed.
SNVS - Secure Non-Volatile Storage		
SNVS_PMIC_ON_REQ	PMIC_ON_REQ	Not multiplexed.
SNVS_TAMPER	TAMPER	Not multiplexed.
SPDC - SiPix Display Controller		
SPDC_CL	EPDC_SDCLK (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCLK
SPDC_DATA00	EPDC_D0 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA00
SPDC_DATA01	EPDC_D1 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA01
SPDC_DATA02	EPDC_D2 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA02
SPDC_DATA03	EPDC_D3 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA03
SPDC_DATA04	EPDC_D4 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA04
SPDC_DATA05	EPDC_D5 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA05
SPDC_DATA06	EPDC_D6 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA06
SPDC_DATA07	EPDC_D7 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA07
SPDC_DATA08	EPDC_D8 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA08
SPDC_DATA09	EPDC_D9 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA09
SPDC_DATA10	EPDC_D10 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA10
SPDC_DATA11	EPDC_D11 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA11
SPDC_DATA12	EPDC_D12 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA12
SPDC_DATA13	EPDC_D13 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA13
SPDC_DATA14	EPDC_D14 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA14
SPDC_DATA15	EPDC_D15 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA15

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
SPDC_LD	EPDC_SDLE (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDLE
SPDC_RL	EPDC_BDR0 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR0
SPDC_UD	EPDC_BDR1 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR1
SPDC_VCOM0	EPDC_VCOM0 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM0
SPDC_VCOM1	EPDC_VCOM1 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM1
SPDC_XDIOL	EPDC_SDOE (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDOE
SPDC_XDIOR	EPDC_SDOE (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDOE
	EPDC_SDSHR (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDSHR
SPDC_YCKL	EPDC_GDCLK (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDCLK
	EPDC_PWRCTRL0 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL0
SPDC_YCKR	EPDC_GDCLK (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDCLK
	EPDC_SDCE0 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE0
SPDC_YDIODL	EPDC_GDSP (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDSP
	EPDC_PWRCTRL3 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL3
SPDC_YDIODR	EPDC_GDSP (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDSP
	EPDC_SDCE3 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE3
SPDC_YDIOUL	EPDC_GDRL (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDRL
	EPDC_PWRCTRL2 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL2
SPDC_YDIOUR	EPDC_GDRL (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDRL
	EPDC_SDCE2 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE2
SPDC_YOEL	EPDC_GDOE (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDOE
	EPDC_PWRCTRL1 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL1
SPDC_YOER	EPDC_GDOE (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDOE
	EPDC_SDCE1 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE1
<b>SPDIF - Sony/Philips Digital Interface</b>		
SPDIF_EXT_CLK	AUD_MCLK (ALT6)	IOMUXC_SW_MUX_CTL_PAD_AUD_MCLK IOMUXC_SPDIF_TX_CLK2_SELECT_INPUT
	ECSPi2_SCLK (ALT1)	IOMUXC_SW_MUX_CTL_PAD_ECSPi2_SCLK IOMUXC_SPDIF_TX_CLK2_SELECT_INPUT
	FEC_REF_CLK (ALT6)	IOMUXC_SW_MUX_CTL_PAD_FEC_REF_CLK IOMUXC_SPDIF_TX_CLK2_SELECT_INPUT
SPDIF_IN	FEC_TX_EN (ALT2)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_EN IOMUXC_SPDIF_SPDIF_IN1_SELECT_INPUT
	I2C2_SCL (ALT2)	IOMUXC_SW_MUX_CTL_PAD_I2C2_SCL IOMUXC_SPDIF_SPDIF_IN1_SELECT_INPUT
	SD2_DAT5 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA5 IOMUXC_SPDIF_SPDIF_IN1_SELECT_INPUT

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
SPDIF_OUT	FEC_TXD1 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA1
	I2C2_SDA (ALT2)	IOMUXC_SW_MUX_CTL_PAD_I2C2_SDA
	SD2_DAT4 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA4
	SD2_RST (ALT3)	IOMUXC_SW_MUX_CTL_PAD_SD2_RESET
SRC - System Reset Controller		
SRC_BOOT_CFG00	LCD_DAT0 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA00
SRC_BOOT_CFG01	LCD_DAT1 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA01
SRC_BOOT_CFG02	LCD_DAT2 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA02
SRC_BOOT_CFG03	LCD_DAT3 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA03
SRC_BOOT_CFG04	LCD_DAT4 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA04
SRC_BOOT_CFG05	LCD_DAT5 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA05
SRC_BOOT_CFG06	LCD_DAT6 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA06
SRC_BOOT_CFG07	LCD_DAT7 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA07
SRC_BOOT_CFG08	LCD_DAT8 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA08
SRC_BOOT_CFG09	LCD_DAT9 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA09
SRC_BOOT_CFG10	LCD_DAT10 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA10
SRC_BOOT_CFG11	LCD_DAT11 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA11
SRC_BOOT_CFG12	LCD_DAT12 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA12
SRC_BOOT_CFG13	LCD_DAT13 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA13
SRC_BOOT_CFG14	LCD_DAT14 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA14
SRC_BOOT_CFG15	LCD_DAT15 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA15
SRC_BOOT_CFG24	LCD_DAT16 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA16
SRC_BOOT_CFG25	LCD_DAT17 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA17
SRC_BOOT_CFG26	LCD_DAT18 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA18
SRC_BOOT_CFG27	LCD_DAT19 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA19
SRC_BOOT_CFG28	LCD_DAT20 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA20
SRC_BOOT_CFG29	LCD_DAT21 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA21
SRC_BOOT_CFG30	LCD_DAT22 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA22
SRC_BOOT_CFG31	LCD_DAT23 (ALT7)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA23
SRC_BOOT_MODE0	BOOT_MODE0	Not multiplexed.
SRC_BOOT_MODE1	BOOT_MODE1	Not multiplexed.
SRC_ONOFF	ONOFF	Not multiplexed.
SRC_POR_B	POR_B	Not multiplexed.
UART1 - Universal Asynchronous Receiver/Transmitter		
UART1_CTS_B	I2C1_SDA (ALT1)	IOMUXC_SW_MUX_CTL_PAD_I2C1_SDA
		IOMUXC_UART1_UART_RTS_B_SELECT_INPUT
UART1_RTS_B	I2C1_SCL (ALT1)	IOMUXC_SW_MUX_CTL_PAD_I2C1_SCL
		IOMUXC_UART1_UART_RTS_B_SELECT_INPUT

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
UART1_RX_DATA	UART1_RXD (ALT0)	IOMUXC_SW_MUX_CTL_PAD_UART1_RXD IOMUXC_UART1_UART_RX_DATA_SELECT_INPUT
UART1_TX_DATA	UART1_TXD (ALT0)	IOMUXC_SW_MUX_CTL_PAD_UART1_TXD IOMUXC_UART1_UART_RX_DATA_SELECT_INPUT
<b>UART2 - Universal Asynchronous Receiver/Transmitter</b>		
UART2_CTS_B	EPDC_D15 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA15 IOMUXC_UART2_UART_RTS_B_SELECT_INPUT
	LCD_RESET (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_RESET IOMUXC_UART2_UART_RTS_B_SELECT_INPUT
	SD2_DAT7 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA7 IOMUXC_UART2_UART_RTS_B_SELECT_INPUT
UART2_RTS_B	EPDC_D14 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA14 IOMUXC_UART2_UART_RTS_B_SELECT_INPUT
	LCD_VSYNC (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_VSYNC IOMUXC_UART2_UART_RTS_B_SELECT_INPUT
	SD2_DAT6 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA6 IOMUXC_UART2_UART_RTS_B_SELECT_INPUT
UART2_RX_DATA	EPDC_D12 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA12 IOMUXC_UART2_UART_RX_DATA_SELECT_INPUT
	LCD_ENABLE (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_ENABLE IOMUXC_UART2_UART_RX_DATA_SELECT_INPUT
	SD2_DAT4 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA4 IOMUXC_UART2_UART_RX_DATA_SELECT_INPUT
UART2_TX_DATA	EPDC_D13 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA13 IOMUXC_UART2_UART_RX_DATA_SELECT_INPUT
	LCD_HSYNC (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_HSYNC IOMUXC_UART2_UART_RX_DATA_SELECT_INPUT
	SD2_DAT5 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA5 IOMUXC_UART2_UART_RX_DATA_SELECT_INPUT
<b>UART3 - Universal Asynchronous Receiver/Transmitter</b>		
UART3_CTS_B	ECSPI2_SS0 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_ECSPI2_SS0 IOMUXC_UART3_UART_RTS_B_SELECT_INPUT
	EPDC_BDR1 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR1 IOMUXC_UART3_UART_RTS_B_SELECT_INPUT
UART3_RTS_B	ECSPI2_MISO (ALT2)	IOMUXC_SW_MUX_CTL_PAD_ECSPI2_MISO IOMUXC_UART3_UART_RTS_B_SELECT_INPUT
	EPDC_BDR0 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR0 IOMUXC_UART3_UART_RTS_B_SELECT_INPUT

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
UART3_RX_DATA	AUD_RXFS (ALT2)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXFS IOMUXC_UART3_UART_RX_DATA_SELECT_INPUT
	ECSPI2_SCLK (ALT2)	IOMUXC_SW_MUX_CTL_PAD_ECSPI2_SCLK IOMUXC_UART3_UART_RX_DATA_SELECT_INPUT
	EPDC_VCOM0 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM0 IOMUXC_UART3_UART_RX_DATA_SELECT_INPUT
UART3_TX_DATA	AUD_RXC (ALT2)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXC IOMUXC_UART3_UART_RX_DATA_SELECT_INPUT
	ECSPI2_MOSI (ALT2)	IOMUXC_SW_MUX_CTL_PAD_ECSPI2_MOSI IOMUXC_UART3_UART_RX_DATA_SELECT_INPUT
	EPDC_VCOM1 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM1 IOMUXC_UART3_UART_RX_DATA_SELECT_INPUT
UART4 - Universal Asynchronous Receiver/Transmitter		
UART4_CTS_B	AUD_TXD (ALT2)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXD IOMUXC_UART4_UART_RTS_B_SELECT_INPUT
	KEY_ROW7 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW7 IOMUXC_UART4_UART_RTS_B_SELECT_INPUT
	SD1_DAT7 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA7 IOMUXC_UART4_UART_RTS_B_SELECT_INPUT
UART4_RTS_B	AUD_TXFS (ALT2)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXFS IOMUXC_UART4_UART_RTS_B_SELECT_INPUT
	KEY_COL7 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL7 IOMUXC_UART4_UART_RTS_B_SELECT_INPUT
	SD1_DAT6 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA6 IOMUXC_UART4_UART_RTS_B_SELECT_INPUT
UART4_RX_DATA	AUD_RXD (ALT2)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXD IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT
	KEY_COL6 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL6 IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT
	SD1_DAT4 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA4 IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT
	UART1_RXD (ALT2)	IOMUXC_SW_MUX_CTL_PAD_UART1_RXD IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT

Table continues on the next page...



**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
UART4_TX_DATA	AUD_TXC (ALT2)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXC IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT
	KEY_ROW6 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW6 IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT
	SD1_DAT5 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA5 IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT
	UART1_TXD (ALT2)	IOMUXC_SW_MUX_CTL_PAD_UART1_TXD IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT
UART5 - Universal Asynchronous Receiver/Transmitter		
UART5_CTS_B	ECSPI1_SS0 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_ECSP11_SS0 IOMUXC_UART5_UART_RTS_B_SELECT_INPUT
	LCD_DAT13 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA13 IOMUXC_UART5_UART_RTS_B_SELECT_INPUT
	SD2_DAT1 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA1 IOMUXC_UART5_UART_RTS_B_SELECT_INPUT
UART5_DCD_B	UART1_TXD (ALT7)	IOMUXC_SW_MUX_CTL_PAD_UART1_TXD
UART5_DSR_B	LCD_DAT3 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA03
UART5_DTR_B	LCD_DAT0 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA00
UART5_RI_B	WDOG_B (ALT2)	IOMUXC_SW_MUX_CTL_PAD_WDOG_B
UART5_RTS_B	ECSPI1_MISO (ALT2)	IOMUXC_SW_MUX_CTL_PAD_ECSP11_MISO IOMUXC_UART5_UART_RTS_B_SELECT_INPUT
	LCD_DAT12 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA12 IOMUXC_UART5_UART_RTS_B_SELECT_INPUT
	SD2_DAT0 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA0 IOMUXC_UART5_UART_RTS_B_SELECT_INPUT
UART5_RX_DATA	ECSPI1_SCLK (ALT2)	IOMUXC_SW_MUX_CTL_PAD_ECSP11_SCLK IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT
	LCD_DAT14 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA14 IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT
	SD2_DAT2 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA2 IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT
	UART1_RXD (ALT4)	IOMUXC_SW_MUX_CTL_PAD_UART1_RXD IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
UART5_TX_DATA	ECSPI1_MOSI (ALT2)	IOMUXC_SW_MUX_CTL_PAD_ECSP11_MOSI IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT
	LCD_DAT15 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA15 IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT
	SD2_DAT3 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA3 IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT
	UART1_TXD (ALT4)	IOMUXC_SW_MUX_CTL_PAD_UART1_TXD IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT
USB - Universal Serial Bus Controller		
USB_H_DATA	HSIC_DAT (ALT0)	IOMUXC_SW_MUX_CTL_PAD_USB_H_DATA
USB_H_STROBE	HSIC_STROBE (ALT0)	IOMUXC_SW_MUX_CTL_PAD_USB_H_STROBE
USB_OTG1_DN	USB_OTG1_DN	Not multiplexed.
USB_OTG1_DP	USB_OTG1_DP	Not multiplexed.
USB_OTG1_ID	EPDC_PWRCOM (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_COM IOMUXC_ANALOG_USB_OTG_ID_SELECT_INPUT
	FEC_RXD0 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA0 IOMUXC_ANALOG_USB_OTG_ID_SELECT_INPUT
	LCD_DAT1 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA01 IOMUXC_ANALOG_USB_OTG_ID_SELECT_INPUT
	REF_CLK_32K (ALT3)	IOMUXC_SW_MUX_CTL_PAD_REF_CLK_32K IOMUXC_ANALOG_USB_OTG_ID_SELECT_INPUT
	SD3_DAT0 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA0 IOMUXC_ANALOG_USB_OTG_ID_SELECT_INPUT
USB_OTG1_OC	ECSPI2_MISO (ALT6)	IOMUXC_SW_MUX_CTL_PAD_ECSP12_MISO IOMUXC_USB_OTG1_OC_SELECT_INPUT
	KEY_ROW4 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW4 IOMUXC_USB_OTG1_OC_SELECT_INPUT
	SD3_DAT3 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA3 IOMUXC_USB_OTG1_OC_SELECT_INPUT
USB_OTG1_PWR	ECSPI2_SS0 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_ECSP12_SS0
	KEY_COL4 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL4
	SD3_CLK (ALT6)	IOMUXC_SW_MUX_CTL_PAD_SD3_CLK
USB_OTG2_DN	USB_OTG2_DN	Not multiplexed.
USB_OTG2_DP	USB_OTG2_DP	Not multiplexed.

*Table continues on the next page...*

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
USB_OTG2_ID	EPDC_PWRINT (ALT4)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_IRQ IOMUXC_ANALOG_USB_H1_ID_SELECT_INPUT
	LCD_DAT0 (ALT2)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA00 IOMUXC_ANALOG_USB_H1_ID_SELECT_INPUT
	REF_CLK_24M (ALT3)	IOMUXC_SW_MUX_CTL_PAD_REF_CLK_24M IOMUXC_ANALOG_USB_H1_ID_SELECT_INPUT
	SD3_CMD (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD3_CMD IOMUXC_ANALOG_USB_H1_ID_SELECT_INPUT
USB_OTG2_OC	ECSPI1_SCLK (ALT6)	IOMUXC_SW_MUX_CTL_PAD_ECSP11_SCLK IOMUXC_USB_OTG2_OC_SELECT_INPUT
	ECSPI2_SCLK (ALT6)	IOMUXC_SW_MUX_CTL_PAD_ECSP12_SCLK IOMUXC_USB_OTG2_OC_SELECT_INPUT
	KEY_ROW5 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW5 IOMUXC_USB_OTG2_OC_SELECT_INPUT
	SD3_DAT2 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA2 IOMUXC_USB_OTG2_OC_SELECT_INPUT
USB_OTG2_PWR	ECSPI1_SS0 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_ECSP11_SS0
	KEY_COL5 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL5
	SD3_CMD (ALT6)	IOMUXC_SW_MUX_CTL_PAD_SD3_CMD
USB_OTG_CHD_B	USB_OTG_CHD_B	Not multiplexed.
USDHC1 - Ultra Secured Digital Host Controller		
SD1_CD_B	ECSPI2_SS0 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_ECSP12_SS0 IOMUXC_USDHC1_CARD_DET_SELECT_INPUT
	FEC_TXD1 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA1 IOMUXC_USDHC1_CARD_DET_SELECT_INPUT
	KEY_COL0 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL0 IOMUXC_USDHC1_CARD_DET_SELECT_INPUT
	KEY_ROW7 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW7 IOMUXC_USDHC1_CARD_DET_SELECT_INPUT
SD1_CLK	SD1_CLK (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD1_CLK
SD1_CMD	SD1_CMD (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD1_CMD
SD1_DATA0	SD1_DAT0 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA0
SD1_DATA1	SD1_DAT1 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA1
SD1_DATA2	SD1_DAT2 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA2
SD1_DATA3	SD1_DAT3 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA3
SD1_DATA4	SD1_DAT4 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA4
SD1_DATA5	SD1_DAT5 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA5
SD1_DATA6	SD1_DAT6 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA6
SD1_DATA7	SD1_DAT7 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD1_DATA7

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
SD1_LCTL	AUD_RXD (ALT4)	IOMUXC_SW_MUX_CTL_PAD_AUD_RXD
	REF_CLK_32K (ALT4)	IOMUXC_SW_MUX_CTL_PAD_REF_CLK_32K
SD1_RESET	ECSPI2_SCLK (ALT4)	IOMUXC_SW_MUX_CTL_PAD_ECSPi2_SCLK
	FEC_MDC (ALT3)	IOMUXC_SW_MUX_CTL_PAD_FEC_MDC
	KEY_COL3 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL3
SD1_VSELECT	ECSPi2_MOSI (ALT4)	IOMUXC_SW_MUX_CTL_PAD_ECSPi2_MOSI
	FEC_RXD0 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA0
	KEY_ROW3 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW3
	SD3_DAT1 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA1
SD1_WP	ECSPi2_MISO (ALT4)	IOMUXC_SW_MUX_CTL_PAD_ECSPi2_MISO IOMUXC_USDHC1_WP_ON_SELECT_INPUT
	FEC_TX_EN (ALT3)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_EN IOMUXC_USDHC1_WP_ON_SELECT_INPUT
	KEY_COL7 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL7 IOMUXC_USDHC1_WP_ON_SELECT_INPUT
	KEY_ROW0 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW0 IOMUXC_USDHC1_WP_ON_SELECT_INPUT
USDHC2 - Ultra Secured Digital Host Controller		
SD2_CD_B	ECSPi1_SS0 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_ECSPi1_SS0 IOMUXC_USDHC2_CARD_DET_SELECT_INPUT
	EPDC_GDSP (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDSP IOMUXC_USDHC2_CARD_DET_SELECT_INPUT
	SD2_DAT7 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA7 IOMUXC_USDHC2_CARD_DET_SELECT_INPUT
SD2_CLK	SD2_CLK (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD2_CLK
SD2_CMD	SD2_CMD (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD2_CMD
SD2_DATA0	SD2_DAT0 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA0
SD2_DATA1	SD2_DAT1 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA1
SD2_DATA2	SD2_DAT2 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA2
SD2_DATA3	SD2_DAT3 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA3
SD2_DATA4	SD2_DAT4 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA4
SD2_DATA5	SD2_DAT5 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA5
SD2_DATA6	SD2_DAT6 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA6
SD2_DATA7	SD2_DAT7 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA7
SD2_LCTL	AUD_TXC (ALT4)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXC
SD2_RESET	ECSPi1_SCLK (ALT4)	IOMUXC_SW_MUX_CTL_PAD_ECSPi1_SCLK
	EPDC_GDCLK (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDCLK
	SD2_RST (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD2_RESET

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
SD2_VSELECT	ECSPI1_MOSI (ALT4)	IOMUXC_SW_MUX_CTL_PAD_ECSPI1_MOSI
	EPDC_GDOE (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDOE
SD2_WP	ECSPI1_MISO (ALT4)	IOMUXC_SW_MUX_CTL_PAD_ECSPI1_MISO
		IOMUXC_USDHC2_WP_ON_SELECT_INPUT
	EPDC_GDRL (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_GDRL
		IOMUXC_USDHC2_WP_ON_SELECT_INPUT
SD2_DAT6 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA6	
		IOMUXC_USDHC2_WP_ON_SELECT_INPUT
USDHC3 - Ultra Secured Digital Host Controller		
SD3_CD_B	EPDC_PWRWAKEUP (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_WAKE
		IOMUXC_USDHC3_CARD_DET_SELECT_INPUT
	FEC_TXD1 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA1
		IOMUXC_USDHC3_CARD_DET_SELECT_INPUT
	I2C2_SDA (ALT4)	IOMUXC_SW_MUX_CTL_PAD_I2C2_SDA
		IOMUXC_USDHC3_CARD_DET_SELECT_INPUT
	REF_CLK_32K (ALT6)	IOMUXC_SW_MUX_CTL_PAD_REF_CLK_32K
		IOMUXC_USDHC3_CARD_DET_SELECT_INPUT
SD3_CLK	SD3_CLK (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD3_CLK
SD3_CMD	SD3_CMD (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD3_CMD
SD3_DATA0	SD3_DAT0 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA0
SD3_DATA1	SD3_DAT1 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA1
SD3_DATA2	SD3_DAT2 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA2
SD3_DATA3	SD3_DAT3 (ALT0)	IOMUXC_SW_MUX_CTL_PAD_SD3_DATA3
SD3_DATA4	KEY_COL1 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL1
		IOMUXC_USDHC3_DATA4_IN_SELECT_INPUT
	SD2_DAT4 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA4
		IOMUXC_USDHC3_DATA4_IN_SELECT_INPUT
SD3_DATA5	KEY_ROW1 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW1
		IOMUXC_USDHC3_DATA5_IN_SELECT_INPUT
	SD2_DAT5 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA5
		IOMUXC_USDHC3_DATA5_IN_SELECT_INPUT
SD3_DATA6	KEY_COL2 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL2
		IOMUXC_USDHC3_DATA6_IN_SELECT_INPUT
	SD2_DAT6 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA6
		IOMUXC_USDHC3_DATA6_IN_SELECT_INPUT

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
SD3_DATA7	KEY_ROW2 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW2 IOMUXC_USDHC3_DATA7_IN_SELECT_INPUT
	SD2_DAT7 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_SD2_DATA7 IOMUXC_USDHC3_DATA7_IN_SELECT_INPUT
SD3_LCTL	AUD_TXFS (ALT4)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXFS
SD3_RESET	EPDC_PWRCOM (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_COM
	FEC_MDC (ALT4)	IOMUXC_SW_MUX_CTL_PAD_FEC_MDC
	I2C1_SCL (ALT4)	IOMUXC_SW_MUX_CTL_PAD_I2C1_SCL
	KEY_COL6 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL6
SD3_VSELECT	EPDC_PWRINT (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_IRQ
	FEC_RXD0 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA0
	I2C1_SDA (ALT4)	IOMUXC_SW_MUX_CTL_PAD_I2C1_SDA
	KEY_ROW6 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW6
SD3_WP	EPDC_PWRSTAT (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_STAT IOMUXC_USDHC3_WP_ON_SELECT_INPUT
	FEC_TX_EN (ALT4)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_EN IOMUXC_USDHC3_WP_ON_SELECT_INPUT
	I2C2_SCL (ALT4)	IOMUXC_SW_MUX_CTL_PAD_I2C2_SCL IOMUXC_USDHC3_WP_ON_SELECT_INPUT
	REF_CLK_24M (ALT6)	IOMUXC_SW_MUX_CTL_PAD_REF_CLK_24M IOMUXC_USDHC3_WP_ON_SELECT_INPUT
<b>USDHC4 - Ultra Secured Digital Host Controller</b>		
SD4_CD_B	EPDC_D11 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA11 IOMUXC_USDHC4_CARD_DET_SELECT_INPUT
	EPDC_PWRCTRL3 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL3 IOMUXC_USDHC4_CARD_DET_SELECT_INPUT
SD4_CLK	EPDC_BDR0 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR0 IOMUXC_USDHC4_CARD_CLK_IN_SELECT_INPUT
	FEC_MDIO (ALT1)	IOMUXC_SW_MUX_CTL_PAD_FEC_MDIO IOMUXC_USDHC4_CARD_CLK_IN_SELECT_INPUT
	KEY_COL4 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL4 IOMUXC_USDHC4_CARD_CLK_IN_SELECT_INPUT
SD4_CMD	EPDC_BDR1 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR1 IOMUXC_USDHC4_CMD_IN_SELECT_INPUT
	FEC_TX_CLK (ALT1)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_CLK IOMUXC_USDHC4_CMD_IN_SELECT_INPUT
	KEY_ROW4 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW4 IOMUXC_USDHC4_CMD_IN_SELECT_INPUT

Table continues on the next page...

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
SD4_DATA0	EPDC_PWRCOM (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_COM IOMUXC_USDHC4_DATA0_IN_SELECT_INPUT
	FEC_RX_ER (ALT1)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_ER IOMUXC_USDHC4_DATA0_IN_SELECT_INPUT
	KEY_COL5 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL5 IOMUXC_USDHC4_DATA0_IN_SELECT_INPUT
SD4_DATA1	EPDC_PWRINT (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_IRQ IOMUXC_USDHC4_DATA1_IN_SELECT_INPUT
	FEC_CRSDV (ALT1)	IOMUXC_SW_MUX_CTL_PAD_FEC_CRSDV IOMUXC_USDHC4_DATA1_IN_SELECT_INPUT
	KEY_ROW5 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW5 IOMUXC_USDHC4_DATA1_IN_SELECT_INPUT
SD4_DATA2	EPDC_PWRSTAT (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_STAT IOMUXC_USDHC4_DATA2_IN_SELECT_INPUT
	FEC_RXD1 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA1 IOMUXC_USDHC4_DATA2_IN_SELECT_INPUT
	KEY_COL6 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL6 IOMUXC_USDHC4_DATA2_IN_SELECT_INPUT
SD4_DATA3	EPDC_PWRWAKEUP (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_WAKE IOMUXC_USDHC4_DATA3_IN_SELECT_INPUT
	FEC_TXD0 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA0 IOMUXC_USDHC4_DATA3_IN_SELECT_INPUT
	KEY_ROW6 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW6 IOMUXC_USDHC4_DATA3_IN_SELECT_INPUT
SD4_DATA4	FEC_MDC (ALT1)	IOMUXC_SW_MUX_CTL_PAD_FEC_MDC IOMUXC_USDHC4_DATA4_IN_SELECT_INPUT
	KEY_COL7 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL7 IOMUXC_USDHC4_DATA4_IN_SELECT_INPUT
	LCD_CLK (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_CLK IOMUXC_USDHC4_DATA4_IN_SELECT_INPUT
SD4_DATA5	FEC_RXD0 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA0 IOMUXC_USDHC4_DATA5_IN_SELECT_INPUT
	KEY_ROW7 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW7 IOMUXC_USDHC4_DATA5_IN_SELECT_INPUT
	LCD_ENABLE (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_ENABLE IOMUXC_USDHC4_DATA5_IN_SELECT_INPUT

*Table continues on the next page...*

**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
SD4_DATA6	FEC_TX_EN (ALT1)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_EN IOMUXC_USDHC4_DATA6_IN_SELECT_INPUT
	KEY_COL3 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_KEY_COL3 IOMUXC_USDHC4_DATA6_IN_SELECT_INPUT
	LCD_HSYNC (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_HSYNC IOMUXC_USDHC4_DATA6_IN_SELECT_INPUT
SD4_DATA7	FEC_TXD1 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA1 IOMUXC_USDHC4_DATA7_IN_SELECT_INPUT
	KEY_ROW3 (ALT4)	IOMUXC_SW_MUX_CTL_PAD_KEY_ROW3 IOMUXC_USDHC4_DATA7_IN_SELECT_INPUT
	LCD_VSYNC (ALT1)	IOMUXC_SW_MUX_CTL_PAD_LCD_VSYNC IOMUXC_USDHC4_DATA7_IN_SELECT_INPUT
SD4_LCTL	AUD_TXD (ALT4)	IOMUXC_SW_MUX_CTL_PAD_AUD_TXD
SD4_RESET	EPDC_D8 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA08
	EPDC_PWRCTRL0 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL0
	FEC_REF_CLK (ALT1)	IOMUXC_SW_MUX_CTL_PAD_FEC_REF_CLK
SD4_VSELECT	EPDC_D9 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA09
	EPDC_PWRCTRL1 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL1
SD4_WP	EPDC_D10 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA10 IOMUXC_USDHC4_WP_ON_SELECT_INPUT
	EPDC_PWRCTRL2 (ALT6)	IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL2 IOMUXC_USDHC4_WP_ON_SELECT_INPUT
<b>WDOG1 - Watchdog Timer</b>		
WDOG1_B	FEC_REF_CLK (ALT2)	IOMUXC_SW_MUX_CTL_PAD_FEC_REF_CLK
	WDOG_B (ALT0)	IOMUXC_SW_MUX_CTL_PAD_WDOG_B
WDOG1_RESET_B_DEB	SD3_CLK (ALT4)	IOMUXC_SW_MUX_CTL_PAD_SD3_CLK
	WDOG_B (ALT1)	IOMUXC_SW_MUX_CTL_PAD_WDOG_B
<b>WDOG2 - Watchdog Timer</b>		
WDOG2_B	EPDC_SDCE1 (ALT1)	IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE1
	SD2_RST (ALT2)	IOMUXC_SW_MUX_CTL_PAD_SD2_RESET
WDOG2_RESET_B_DEB	AUD_MCLK (ALT4)	IOMUXC_SW_MUX_CTL_PAD_AUD_MCLK
	LCD_DAT4 (ALT3)	IOMUXC_SW_MUX_CTL_PAD_LCD_DATA04
<b>XTALOSC - Crystal Oscillator</b>		
XTALOSC_CLK1_N	CLK1_N	Not multiplexed.
XTALOSC_CLK1_P	CLK1_P	Not multiplexed.
XTALOSC_REF_CLK_24M	REF_CLK_24M (ALT0)	IOMUXC_SW_MUX_CTL_PAD_REF_CLK_24M
	HSIC_DAT (ALT3)	IOMUXC_SW_MUX_CTL_PAD_USB_H_DATA
XTALOSC_REF_CLK_32K	REF_CLK_32K (ALT0)	IOMUXC_SW_MUX_CTL_PAD_REF_CLK_32K
	HSIC_STROBE (ALT3)	IOMUXC_SW_MUX_CTL_PAD_USB_H_STROBE

Table continues on the next page...



**Table 4-2. Muxing Options (continued)**

Signal	Pad (Mode)	Mux/Input Select Registers
XTALOSC_RTC_XTALI	RTC_XTALI	Not multiplexed.
XTALOSC_RTC_XTALO	RTC_XTALO	Not multiplexed.
XTALOSC_XTALI	XTALI	Not multiplexed.
XTALOSC_XTALO	XTALO	Not multiplexed.



# Chapter 5

## Fusemap

### 5.1 Boot Fusemap

The following section details the various modes and selection of the required boot devices.

A separate map is given for each and every boot device. The device select is specified by BOOT\_CFG1[6:4] fuses listed in [Table 5-1](#).

**Table 5-1. Boot Device Select**

Boot Device	BOOT_CFG1[6]	BOOT_CFG1[5]	BOOT_CFG1[4]
EIM ( <a href="#">Table 5-2</a> )	0	0	0
Serial ROM ( <a href="#">Table 5-3</a> )	0	1	1
SD/eSD ( <a href="#">Table 5-4</a> )	1	0	X
MMC/eMMC ( <a href="#">Table 5-5</a> )	1	1	X

#### NOTE

Fuses marked as “Reserved” are reserved for Freescale internal (and future) use only. Customers should not attempt to burn these, as the IC behavior may be unpredictable. The reserved fuses can be read as either '0' or '1'.

**Table 5-2. EIM Boot Fusemap**

Addr	7	6	5	4	3	2	1	0
0x450[7:0] (BOOT_CFG1)	Use L2 Cache as OCRAM  0 - L2 Cache enabled mode  1 - L2 Cache disabled and L2 Cache memory will be used as OCRAM	0	0	0	Memory Type:  0 - NOR Flash  1 - OneNAND	Reserved	Reserved	Reserved
0x450[15:8] (BOOT_CFG2)	Muxing Scheme:  00 - A/D16 (HW Default in external boot)  01 - A+DH  10 - A+DL  11 - Reserved		OneNand Page Size:  00 - 1KB  01 - 2KB  10 - 4KB  11 - Reserved		Reserved	Boot Frequencies (ARM/DDR)  0 - 792 / 400 MHz  1 - 528 / 307 MHz	Reserved	EIM VOLTAGE SELECTIO N  0 - 3.3V  1 - 1.8V
0x450[23:16] (BOOT_CFG3)	Reserved							
0x450[31:24] (BOOT_CFG4)	Reserved	EEPROM Recovery Enable  0 - Disabled  1 - Enabled Fuse is Reserved for 'Serial- ROM' Boot mode	eCSPI chip select:  00 - ECSPiX_SS0 (default)  01 - ECSPiX_SS1  10 - ECSPiX_SS2  11 - ECSPiX_SS3		eCSPI Addressing:  0 - 2-bytes (16-bit)  1 - 3-bytes (24-bit)	Port Select:  000 - eCSPI1  001 - eCSPI2  010 - eCSPI3  011 - eCSPI4  100 - eCSPI5  101 - I2C1  110 - I2C2  111 - I2C3		
0x460[7:0]	Reserved			BT_FUSE_ SEL	DIR_BT_DI S	Reserved	SEC_CONF IG[1]	Reserved
0x460[15:8]	Reserved							
0x460[23:16]	JTAG_SMODE[1:0]		WDOG_EN ABLE  0 - Disabled  1 - Enabled	SJC_DISAB LE	Reserved			
0x460[31:24]	Reserved			TZASC_EN ABLE	JTAG_HEO	KTE	Reserved	

Table continues on the next page...

**Table 5-2. EIM Boot Fusemap (continued)**

Addr	7	6	5	4	3	2	1	0
0x470[7:0]	DLL Override:  0 - Boot ROM default  1 - Apply value per fuse field MMC_DLL_ DLY[3:0]	SD4 VOLTAGE SELECTIO N  0 - 3.3V  1 - 1.8V	SD3 VOLTAGE SELECTIO N  0 - 3.3V  1 - 1.8V	SD1 VOLTAGE SELECTIO N  0 - 3.3V  1 - 1.8V	Disable SDMMC Manufactur e mode  0 - Enable  1 - Disable	L1 I-Cache DISABLE	BT_MMU_D ISABLE	Override SD Pad Settings  (using PAD_SETTI NGS value)
0x470[15:8]	Reserved	eMMC_ RESET_EN	SDMMC_ HYS_EN	USDHC_PA D_PULL_D OWN  0 - no action  1 - pull down	ENABLE_E MMC_22K_ PULLUP  0 - 47K pullup  1 - 22K pullup	ADD_DS_ SET_GPR1 _16  0 - Set  1 - Don't set	USDHC_IO MUX_SION _BIT_ENAB LE  0 - Disable  1 - Enable	USDHC IOMUX SRE Enable  0 - Disable  1 - Enable
0x470[23:16]	Reserved	LPB_BOOT (Core / DDR-Bus)  00 - LPB Disable  01 - 1 GPIO (def freq)  10 - Div by2  11 - Div by 4		BT_LPB_P OLARITY  (GPIO polarity)  0 - Active High  1 -Active Low	Reserved			
0x470[31:24]	Reserved	MMC_DLL_DLY[6:0]						
0x6D0[7:0]	Reserved (locked by MISC_CONF_LOCK)		PAD_SETTINGS[5:0]					

**Table 5-3. Serial-ROM Boot Fusemap**

Addr	7	6	5	4	3	2	1	0
0x450[7:0] (BOOT_CFG1)	Use L2 Cache as OCRAM 0 - L2 Cache enabled mode 1 - L2 Cache disabled and L2 Cache memory will be used as OCRAM	0	1	1	Reserved	Reserved	Reserved	Reserved

Table continues on the next page...

Table 5-3. Serial-ROM Boot Fusemap (continued)

Addr	7	6	5	4	3	2	1	0
0x450[15:8] (BOOT_CFG2)	Reserved	Reserved	Reserved	Reserved	Reserved	Boot Frequencies (ARM/DDR)  0 - 792 / 400 MHz  1 - 528 / 307 MHz	Reserved	SPI/I2C VOLTAGE SELECTIO N  0 - 3.3V  1 - 1.8V
0x450[23:16] (BOOT_CFG3)	Reserved							
0x450[31:24] (BOOT_CFG4)	Reserved	EEPROM Recovery Enable  0 - Disabled  1 - Enabled Fuse is Reserved for 'Serial- ROM' Boot mode	eCSPI chip select:  00 - ECSPiX_SS0 (default)  01 - ECSPiX_SS1  10 - ECSPiX_SS2  11 - ECSPiX_SS3	eCSPI Addressing:  0 - 2-bytes (16-bit)  1 - 3-bytes (24-bit)	Port Select:  000 - eCSPI1  001 - eCSPI2  010 - eCSPI3  011 - eCSPI4  100 - eCSPI5  101 - I2C1  110 - I2C2  111 - I2C3			
0x460[7:0]	Reserved			BT_FUSE_ SEL	DIR_BT_DI S	Reserved	SEC_CONF IG[1]	Reserved
0x460[15:8]	Reserved							
0x460[23:16]	JTAG_SMODE[1:0]		WDOG_EN ABLE  0 - Disabled  1 - Enabled	SJC_DISAB LE	Reserved			
0x460[31:24]	Reserved			TZASC_EN ABLE	JTAG_HEO	KTE	Reserved	
0x470[7:0]	DLL Override:  0 - Boot ROM default  1 - Apply value per fuse field MMC_DLL_ DLY[3:0]	SD4 VOLTAGE SELECTIO N  0 - 3.3V  1 - 1.8V	SD3 VOLTAGE SELECTIO N  0 - 3.3V  1 - 1.8V	SD1 VOLTAGE SELECTIO N  0 - 3.3V  1 - 1.8V	Disable SDMMC Manufactur e mode  0 - Enable  1 - Disable	L1 I-Cache DISABLE	BT_MMU_D ISABLE	Override SD Pad Settings  (using PAD_SETTI NGS value)
0x470[15:8]	Reserved	eMMC_ RESET_EN	SDMMC_ HYS_EN	USDHC_PA D_PULL_D OWN  0 - no action  1 - pull down	ENABLE_E MMC_22K_ PULLUP  0 - 47K pullup  1 - 22K pullup	ADD_DS_ SET_GPR1 _16  0 - Set  1 - Don't set	USDHC_IO MUX_SION _BIT_ENAB LE  0 - Disable  1 - Enable	USDHC IOMUX SRE Enable  0 - Disable  1 - Enable

Table continues on the next page...

**Table 5-3. Serial-ROM Boot Fusemap (continued)**

Addr	7	6	5	4	3	2	1	0
0x470[23:16]	Reserved	LPB_BOOT (Core / DDR-Bus)  00 - LPB Disable 01 - 1 GPIO (def freq) 10 - Div by2 11 - Div by 4		BT_LPB_POLARITY  (GPIO polarity)  0 - Active High  1 -Active Low	Reserved			
0x470[31:24]	Reserved	MMC_DLL_DLY[6:0]						
0x6D0[7:0]	Reserved (locked by MISC_CONF_LOCK)		PAD_SETTINGS[5:0]					

**Table 5-4. SD/eSD Boot Fusemap**

Addr	7	6	5	4	3	2	1	0
0x450[7:0] (BOOT_CFG1)	Use L2 Cache as OCRAM  0 - L2 Cache enabled mode  1 - L2 Cache disabled and L2 Cache memory will be used as OCRAM	1	0	Fast Boot:  0 - Regular  1 - Fast Boot	SD/SDXC Speed  00 - Normal/SDR12  01 - High/SDR25  10 - SDR50  11 - SDR104		SD Power Cycle Enable  0 - No power cycle  1 - Enabled via USDHC_RS T pad (uSDHC3 & 4 only)	SD Loopback Clock Source Sel (for SDR50 and SDR104 only)  0 - through SD pad  1 - direct
0x450[15:8] (BOOT_CFG2)	SD Calibration Step  00 - 1 delay cell  01 - 1 delay cell  10 - 2 delay cell  11 - 3 delay cell		Bus Width:  0 - 1-bit  1 - 4-bit	Port Select:  00 - uSDHC1  01 - uSDHC2  10 - uSDHC3  11 - uSDHC4		Boot Frequencies (ARM/DDR)  0 - 792 / 400 MHz  1 - 528 / 307 MHz	SD2 VOLTAGE SELECTIO N  0 - 3.3V  1 - 1.8V	Reserved
0x450[23:16] (BOOT_CFG3)	Reserved							

Table continues on the next page...

**Table 5-4. SD/eSD Boot Fusemap (continued)**

Addr	7	6	5	4	3	2	1	0
0x450[31:24] (BOOT_CFG4)	Reserved	EEPROM Recovery Enable  0 - Disabled  1 - Enabled Fuse is Reserved for 'Serial- ROM' Boot mode	eCSPI chip select:  00 - ECSPiX_SS0 (default)  01 - ECSPiX_SS1  10 - ECSPiX_SS2  11 - ECSPiX_SS3		eCSPI Addressing:  0 - 2-bytes (16-bit)  1 - 3-bytes (24-bit)	Port Select:  000 - eCSPI1  001 - eCSPI2  010 - eCSPI3  011 - eCSPI4  100 - eCSPI5  101 - I2C1  110 - I2C2  111 - I2C3		
0x460[7:0]	Reserved			BT_FUSE_ SEL	DIR_BT_DI S	Reserved	SEC_CONF IG[1]	Reserved
0x460[15:8]	Reserved							
0x460[23:16]	JTAG_SMODE[1:0]		WDOG_EN ABLE  0 - Disabled  1 - Enabled	SJC_DISAB LE	Reserved			
0x460[31:24]	Reserved			TZASC_EN ABLE	JTAG_HEO	KTE	Reserved	
0x470[7:0]	DLL Override:  0 - Boot ROM default  1 - Apply value per fuse field MMC_DLL_ DLY[3:0]	SD4 VOLTAGE SELECTIO N  0 - 3.3V  1 - 1.8V	SD3 VOLTAGE SELECTIO N  0 - 3.3V  1 - 1.8V	SD1 VOLTAGE SELECTIO N  0 - 3.3V  1 - 1.8V	Disable SDMMC Manufactur e mode  0 - Enable  1 - Disable	L1 I-Cache DISABLE	BT_MMU_D ISABLE	Override SD Pad Settings  (using PAD_SETTI NGS value)
0x470[15:8]	Reserved	eMMC_ RESET_EN	SDMMC_ HYS_EN	USDHC_PA D_PULL_D OWN  0 - no action  1 - pull down	ENABLE_E MMC_22K_ PULLUP  0 - 47K pullup  1 - 22K pullup	ADD_DS_ SET_GPR1 _16  0 - Set  1 - Don't set	USDHC_IO MUX_SION _BIT_ENAB LE  0 - Disable  1 - Enable	USDHC IOMUX SRE Enable  0 - Disable  1 - Enable
0x470[23:16]	Reserved	LPB_BOOT (Core / DDR-Bus)  00 - LPB Disable  01 - 1 GPIO (def freq)  10 - Div by2  11 - Div by 4		BT_LPB_P OLARITY  (GPIO polarity)  0 - Active High  1 -Active Low	Reserved			
0x470[31:24]	Reserved	MMC_DLL_DLY[6:0]						

Table continues on the next page...



**Table 5-4. SD/eSD Boot Fusemap (continued)**

Addr	7	6	5	4	3	2	1	0
0x6D0[7:0]	Reserved (locked by MISC_CONF_LOCK)		PAD_SETTINGS[5:0]					

**Table 5-5. MMC/eMMC Boot Fusemap**

Addr	7	6	5	4	3	2	1	0
0x450[7:0] (BOOT_CFG1)	Use L2 Cache as OCRAM  0 - L2 Cache enabled mode  1 - L2 Cache disabled and L2 Cache memory will be used as OCRAM	1	1	Fast Boot:  0 - Regular  1 - Fast Boot	SD/MMC Speed  0 - High 1- Normal	Fast Boot Acknowledge Disable:  0 - Boot Ack Enabled  1 - Boot Ack Disabled	SD Power Cycle Enable  0 - No power cycle 1 - Enabled via USDHC_RS T pad (uSDHC3 & 4 only)	SD Loopback Clock Source Sel (for SDR50 and SDR104 only)  0 - through SD pad 1 - direct
0x450[15:8] (BOOT_CFG2)	Bus Width:  000 - 1-bit  001 - 4-bit  010 - 8-bit  101 - 4-bit  DDR (MMC 4.4)  110 - 8-bit DDR (MMC 4.4)  Else - Reserved			Port Select:  00 - uSDHC1  01 - uSDHC2  10 - uSDHC3 (eMMC4.4)  11 - uSDHC4		Boot Frequencies (ARM/DDR)  0 - 792 / 400 MHz  1 - 528 / 307 MHz	SD2 VOLTAGE SELECTION  0 - 3.3V  1 - 1.8V	Reserved
0x450[23:16] (BOOT_CFG3)	Reserved							
0x450[31:24] (BOOT_CFG4)	Reserved	EEPROM Recovery Enable  0 - Disabled  1 - Enabled Fuse is Reserved for 'Serial- ROM' Boot mode	eCSPI chip select:  00 - ECSPiX_SS0 (default)  01 - ECSPiX_SS1  10 - ECSPiX_SS2  11 - ECSPiX_SS3		eCSPI Addressing:  0 - 2-bytes (16-bit)  1 - 3-bytes (24-bit)	Port Select:  000 - eCSPI1  001 - eCSPI2  010 - eCSPI3  011 - eCSPI4  100 - eCSPI5  101 - I2C1  110 - I2C2  111 - I2C3		
0x460[7:0]	Reserved			BT_FUSE_ SEL	DIR_BT_DI S	Reserved	SEC_CONF IG[1]	Reserved

Table continues on the next page...

**Table 5-5. MMC/eMMC Boot Fusemap (continued)**

Addr	7	6	5	4	3	2	1	0
0x460[15:8]	Reserved							
0x460[23:16]	JTAG_SMODE[1:0]		WDOG_EN ABLE  0 - Disabled 1 - Enabled	SJC_DISAB LE	Reserved			
0x460[31:24]	Reserved			TZASC_EN ABLE	JTAG_HEO	KTE	Reserved	
0x470[7:0]	DLL Override:  0 - Boot ROM default  1 - Apply value per fuse field MMC_DLL_ DLY[3:0]	SD4 VOLTAGE SELECTIO N  0 - 3.3V 1 - 1.8V	SD3 VOLTAGE SELECTIO N  0 - 3.3V 1 - 1.8V	SD1 VOLTAGE SELECTIO N  0 - 3.3V 1 - 1.8V	Disable SDMMC Manufactur e mode  0 - Enable 1 - Disable	L1 I-Cache DISABLE	BT_MMU_D ISABLE	Override SD Pad Settings  (using PAD_SETTI NGS value)
0x470[15:8]	Reserved	eMMC_ RESET_EN	SDMMC_ HYS_EN	USDHC_PA D_PULL_D OWN  0 - no action 1 - pull down	ENABLE_E MMC_22K_ PULLUP  0 - 47K pullup 1 - 22K pullup	ADD_DS_ SET_GPR1 _16  0 - Set 1 - Don't set	USDHC_IO MUX_SION _BIT_ENAB LE  0 - Disable 1 - Enable	USDHC IOMUX SRE Enable  0 - Disable 1 - Enable
0x470[23:16]	Reserved	LPB_BOOT (Core / DDR-Bus)  00 - LPB Disable 01 - 1 GPIO (def freq) 10 - Div by2 11 - Div by 4		BT_LPB_P OLARITY  (GPIO polarity)  0 - Active High 1 -Active Low	Reserved			
0x470[31:24]	Reserved	MMC_DLL_DLY[6:0]						
0x6D0[7:0]	Reserved (locked by MISC_CONF_LOCK)		PAD_SETTINGS[5:0]					

## 5.2 Lock Fusemap

Table 5-6 describes the functions of various lock fuses.

**Table 5-6. Lock Fuses**

Addr	7	6	5	4	3	2	1	0
0x400[7:0]	Reserved	SJC_RESP_LOCK WRP,OP,RDP	MEM_TRIM_LOCK 1x - OP x1 - WP		BOOT_CFG_LOCK (Locking rows: 0x450-0x470) 1x - OP x1 - WP		TESTER_LOCK 1x - OP x1 - WP	
00x400[15:8]	Reserved	SRK_LOCK RD,WR,OP	GP2_LOCK 1x - OP x1 - WP		GP1_LOCK 1x - OP x1 - WP		MAC_ADDR_LOCK 1x - OP x1 - WP	
0x400[23:16]	Reserved	MISC_CONF_LOCK 1 - WP + OP of MISC_CONF	Reserved		ANALOG_LOCK 1x - OP x1 - WP		Reserved	SW_GP_LOCK 1 - WP + OP of SW_GP fuses
0x400[31:24]	Reserved		GP_HI_LOCK 1x - OP x1 - WP		GP_LO_LOCK 1x - OP x1 - WP		Reserved	

## 5.3 Fusemap Descriptions Table

This section covers the fusemap descriptions of the chip.

**Table 5-7. Fusemap Descriptions**

Fuse Address	Fuses Name	Number of Fuses	Fuses Function	Settings	Locked by
0x400[1:0]	TESTER_LOCK	2	Provides Locking of various fuse bits.	00 - Unlock (The controlled field can be read, sensed, burned or overridden in the corresponded  1x - Override Protect (OP) x1 - Write Protect (WP) 11 - Both OP and WP	N/A

*Table continues on the next page...*

**Table 5-7. Fusemap Descriptions (continued)**

Fuse Address	Fuses Name	Number of Fuses	Fuses Function	Settings	Locked by
0x400[3:2]	BOOT_CFG_LOCK	2	Perform lock on BOOT related fuses.	00 - Unlock (The controlled field can be read, sensed, burned or overridden in the corresponded 1x - Override Protect (OP) x1 - Write Protect (WP) 11 - Both OP and WP	N/A
0x400[5:4]	MEM_TRIM_LOCK	2	Trimming fuses. Burnt on the tester or by customer before the final product shipment.	00 - Unlock (The controlled field can be read, sensed, burned or overridden in the corresponded. 1x - Override Protect (OP) x1 - Write Protect (WP) 11 - Both OP and WP	N/A
0x400[6]	SJC_RESP_LOCK	1	Lock bit for JTAG response fuses. (SJC_RESP)	00 - Unlock 1 - Read Protect + Write Protect + Override Protect and Program protect lock	N/A
0x400[9:8]	MAC_ADDR_LOCK	2	Lock MAC_ADDR fuses.	00 - Unlock (The controlled field can be read, sensed, burned or overridden in the corresponded. 1x - Override Protect (OP) x1 - Write Protect (WP) 11 - Both OP and WP	N/A
0x400[11:10]	GP1_LOCK	2	Lock for General Purpose fuse register #1 (GP1)	00 - Unlock (The controlled field can be read, sensed, burned or overridden in the corresponded. 1x - Override Protect (OP) x1 - Write Protect (WP) 11 - Both OP and WP	N/A
0x400[13:12]	GP2_LOCK	2	Lock for General Purpose fuse register #2 (GP2)	00 - Unlock (The controlled field can be read, sensed, burned or overridden in the corresponded. 1x - Override Protect (OP) x1 - Write Protect (WP) 11 - Both OP and WP	N/A
0x400[14]	SRK_LOCK	1	Locking SRK_HASH[255:0]	0 - Unlock 1 - Write Protect + Override Protect	N/A

Table continues on the next page...

**Table 5-7. Fusemap Descriptions (continued)**

Fuse Address	Fuses Name	Number of Fuses	Fuses Function	Settings	Locked by
0x400[16]	SW_GP_LOCK	1	Lock for software General Purpose fuse bits. (address 0x680-0x6C0)	0 - Unlock 1 - WP + OP of SW_GP	N/A
0x400[19:18]	ANALOG_LOCK	2	Lock bit for various fuses (Addresses: 0x4D0-0x4F0)	00 - Unlock (The controlled field can be read, sensed, burned or overridden in the corresponded. 1x - Override Protect (OP) x1 - Write Protect (WP) 11 - Both OP and WP	N/A
0x400[22]	MISC_CONF_LOCK	1	Locking of MISC_CONFIG fuses (address 0x6D0)	0 - Unlock 1 - WP + OP of MISC_CONF	N/A
0x400[27:26]	GP_LO_LOCK	2	Lock for low 256 bits of the additional General Purpose fuse register (Address: 0x700-0x770)	00 - Unlock (The controlled field can be read, sensed, burned or overridden in the corresponded. 1x - Override Protect (OP) x1 - Write Protect (WP) 11 - Both OP and WP	N/A
0x400[29:28]	GP_HI_LOCK	2	Lock for high 256 bits of the additional General Purpose fuse register(0x780-0x7F0)	00 - Unlock (The controlled field can be read, sensed, burned or overridden in the corresponded. 1x - Override Protect (OP) x1 - Write Protect (WP) 11 - Both OP and WP	N/A
0x410[31:0]	SJC_CHALL[31:0] / UNIQUE_ID[31:0]	32	Device Unique ID, also be used for SJC Challenge phrase. (bits 31-0)	Random, by test program.	TESTER_LOCK
0x420[31:0]	SJC_CHALL/ UNIQUE_ID[42:32]	32	Device Unique ID, also be used for SJC Challenge phrase. (bits 63-32)	Random, by test program.	TESTER_LOCK
0x430[19:16]	SI_REV[3:0]	4	Silicon Revision number. Can be used in conjunction with HW value set in GPR registers. (gpr4[15:8])	0 - TO 1.0	TESTER_LOCK
0x450[7:0]	BOOT_CFG1	8	BOOT configuration register #1, Usage varies, depending on selected boot device.	See <a href="#">Table 5-1</a> for details.	BOOT_CFG_LOCK
0x450[15:8]	BOOT_CFG2	8	BOOT configuration register #2, Usage varies, depending on selected boot device.	Refer to the respective boot fusemap tables for details.	BOOT_CFG_LOCK
0x450[23:16]	BOOT_CFG3	8	BOOT configuration register #3	Refer to the respective boot fusemap tables for details.	BOOT_CFG_LOCK

Table continues on the next page...

Table 5-7. Fusemap Descriptions (continued)

Fuse Address	Fuses Name	Number of Fuses	Fuses Function	Settings	Locked by
0x450[31:24]	BOOT_CFG4	8	BOOT configuration register #4	Refer to the respective boot fusemap tables for details.	BOOT_CFG_LOCK
0x460[1]	SEC_CONFIG[1]	1	Security Configuration (with SEC_CONFIG[0])	Out of FAB state: 0 - Open - allows any code to be flashed and executed, even if it has no valid signature. 1 - Closed (Security On)	BOOT_CFG_LOCK
0x460[3]	DIR_BT_DIS	1	Direct External Memory Boot Disable	0 - Direct boot from external memory is allowed 1 - Direct boot from external memory is not allowed	BOOT_CFG_LOCK
0x460[4]	BT_FUSE_SEL	1	Determines, whether using fuses for boot configuration, or GPIO /Serial loader.	If boot_mode="00" (Development) 0=Boot mode configuration is taken from GPIOs 1 - Boot mode configuration is taken from fuses If boot_mode="10" (Production) 0 - Boot using Serial Loader (USB) 1 - Boot mode configuration is taken from fuses	BOOT_CFG_LOCK
0x460[15:8]	DDR3_CONFIG[7:0]	8	TBD (DDR3 config options)		BOOT_CFG_LOCK
0x460[20]	SJC_DISABLE	1	Disable/Enable the Secure JTAG Controller module. This fuse is used to create highest JTAG security level, where JTAG is totally blocked.	0 - Secure JTAG Controller is enabled 1 - Secure JTAG Controller is disabled	BOOT_CFG_LOCK
0x460[21]	WDOG_ENABLE	1	Watchdog Enable	Used to specify whether to enable / not watchdog at boot. 0 - Watch-Dog is disabled 1 - Watch-Dog is enabled	BOOT_CFG_LOCK
0x460[23:22]	JTAG_SMODE[1:0]	2	JTAG Security Mode. Controls the security mode of the JTAG debug interface	00 - JTAG enable mode 01 - Secure JTAG mode 11 - No debug mode	BOOT_CFG_LOCK

Table continues on the next page...

Table 5-7. Fusemap Descriptions (continued)

Fuse Address	Fuses Name	Number of Fuses	Fuses Function	Settings	Locked by
0x460[26]	KTE	1	Kill Trace Enable. Enables tracing capability on ETM, and other modules.	0 - Bus tracing is allowed 1 - Bus tracing is allowed in case security state as defined by Secure JTAG allows it (for example, JTAG_ENABLE or NO_DEBUG)	BOOT_CFG_LOCK
0x460[27]	JTAG_HEO	1	JTAG HAB Enable Override. Disallows HAB JTAG enabling. The HAB may normally enable JTAG debugging by means of the HAB_JDE-bit in the OCOTP SCS register. The JTAG_HEO-bit can override this behavior.	0 - HAB may enable JTAG debug access 1 - HAB JTAG enable is overridden (HAB may not enable JTAG debug access)	BOOT_CFG_LOCK
0x460[28]	TZASC_ENABLE	1	TZASC1, 2 enable fuse.	0 - TZASC1, 2 modules left in disable and bypass state. 1 - TZASC1, 2 modules and associated clocks and muxing are enabled by Boot ROM code.	BOOT_CFG_LOCK
0x470[13]	SDMMC_HYS_EN	1	Override HYS bit for SD/MMC pads	0 - Override HYS bit for SD/MMC pads disabled. 1 - Once the fuse blown, the [HYS] bit of IOMUXC_SW_PAD_CTL_PAD_SDx_CLK, IOMUXC_SW_PAD_CTL_PAD_SDx_CMD, IOMUXC_SW_PAD_CTL_PAD_SDx_DAT0-n will be set.	BOOT_CFG_LOCK
0x460[14]	eMMC_RESET_EN	1	eMMC 4.4 - RESET TO PRE-IDLE STATE	0 - the CMD0 with argument 0xf0f0f0 will not be sent to put the eMMC card into pre-IDLE state. 1 - the CMD0 with argument 0xf0f0f0 will be sent to put the eMMC card into pre-IDLE state so that eMMC card's fast boot can work properly.	BOOT_CFG_LOCK
0x470[20]	BT_LPB_POLARITY	1	Define GPIO3 polarity, for determining LPB boot mode.	0 - Active High 1 - Active Low	BOOT_CFG_LOCK

Table continues on the next page...

Table 5-7. Fusemap Descriptions (continued)

Fuse Address	Fuses Name	Number of Fuses	Fuses Function	Settings	Locked by
0x470[22:21]	LPB_BOOT	2	Defined the LowPower Boot options	(Core / DDR- Bus) 00 - LPB Disable 01 - 1 GPIO (def freq) 01 - Div by2 11 - Div by 4	BOOT_CFG_LOCK
0x470[30:24]	MMC_DLL_DLY[6:0]	7	eMMC 4.4/4/41 delay line default value (set by boot rom), used in conjunction with "DLL Override" = 1 (BOOT_CFG3[3])	See RM for details.	BOOT_CFG_LOCK
0x4F0[15:0]	USB_VID[31:0]	16	USB VID value,( to be used by USB software driver).		ANALOG_LOCK
0x4F0[31:16]	USB_PID[31:0]	16	USB PID value ( to be used by USB software driver).		ANALOG_LOCK
0x580-0x5F0	SRK_HASH[255:0]	256	SRK key	0x580-SRK0 0x590-SRK1 0x5F0-SRK2 0x5F0-SRK3 0x5F0-SRK4 0x5F0-SRK5 0x5F0-SRK6 0x5F0-SRK7	SRK_LOCK
0x600[31:0]	SJC_RESP[31:0]	32	Response reference value for the secure JTAG controller	Customer / OEM use.	SJC_RESP_LOCK (locks also for read and explicit sense)
0x610[23:0]	SJC_RESP[55:32]	24	Response reference value for the secure JTAG controller	Customer / OEM use.	SJC_RESP_LOCK (locks also for read and explicit sense)
0x620[31:0]	MAC_ADDR[31:0]	32	Reserved for customers/ software	Customer / OEM use.	MAC_ADDR_LOCK
0x630[15:0]	MAC_ADDR[47:32]	16	Reserved for customers/ software	Customer / OEM use.	MAC_ADDR_LOCK
0x660[31:0]	GP1[31:0]	32	General Purpose fuse register #1		GP1_LOCK
0x670[31:0]	GP2[31:0]	32	General Purpose fuse register #2		GP2_LOCK
0x680-0x6C0	SW_GP[159:0]	160	General Purpose fuse register for software		SW_GP_LOCK

Table continues on the next page...



**Table 5-7. Fusemap Descriptions (continued)**

Fuse Address	Fuses Name	Number of Fuses	Fuses Function	Settings	Locked by
0x6D0[5:0]	PAD_SETTINGS	6	Used with conjunction of MMC/SD/Nand "Override Pad Settings" fuse value, as follows:  0 - Use IO default settings for boot device IO pads 1 - Use "Override" value, as set by this register	IO pads settings of selected boot interface, are override with this fuses, as follows:  [0] - Slew Rate [3:1] Drive Strength [5:4] - Speed Settings. Refer to IO PAD chapter for "Settings" fields value	MISC_CONF_LOCK
0x6E0[0]	FIELD_RETURN	1	Configure device for field return testing. Fuse burning requires CSF command.	0 - Device is in functional / secure mode. 1 - Device is open for 'field-return' testing	FIELD_RETURN_LOCK
0x700-0x770	GP[255:0]	256	Additional General Purpose fuse register		GP_LO_LOCK
0x780-0x7F0	GP[511:256]	256	Additional General Purpose fuse register		GP_HI_LOCK



**Fusemap Descriptions Table**

## Chapter 6

# External Memory Controllers

### 6.1 Overview

This chip has the following external memory interfaces and controllers:

- Multi-mode DDR controller (MMDC)
- EIM-PSRAM/NOR Flash controller

### 6.2 Multi-mode DDR controller (MMDC) overview and feature summary

The MMDC module is a DDR controller that can support several types of DDR memories.

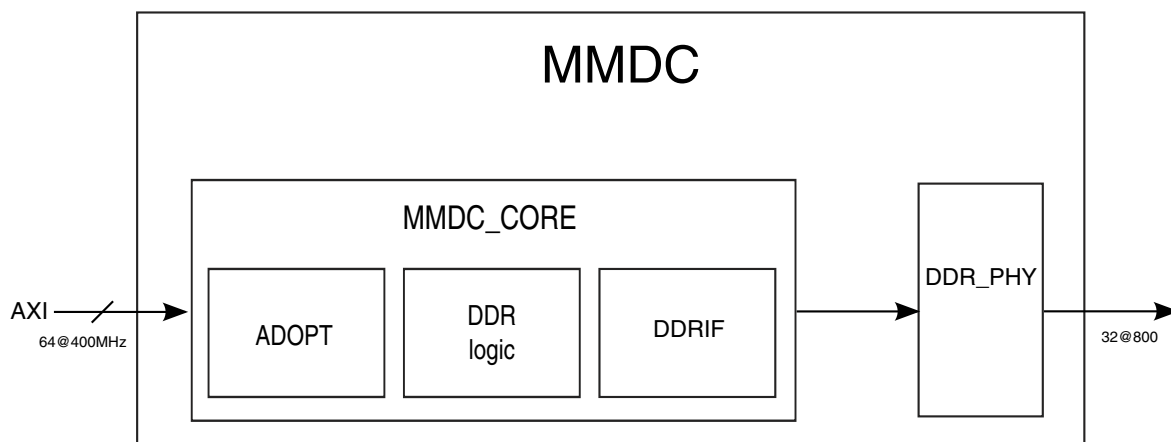


Figure 6-1. MMDC block diagram

**Table 6-1. MMDC feature summary**

Feature	Description
Supported standards	<ul style="list-style-type: none"> <li>• LV-DDR3, DDR3 x16, x32 (includes SODIMM)</li> <li>• LPDDR2 1ch x32</li> </ul>
DDR interface	<ul style="list-style-type: none"> <li>• x16, x32 data bus width</li> <li>• Density of 256 Mbytes-8 Gbytes <ul style="list-style-type: none"> <li>• Column size of 8-12 bits</li> <li>• Row size of 11-16 bits</li> </ul> </li> <li>• Up to 2 Gbyte address space and configurable address space per CS.</li> <li>• Supports burst length of 8 (aligned) for DDR3 and burst lengths of 4 for LPDDR2</li> </ul>
DDR performance	<ul style="list-style-type: none"> <li>• DDR3 and LPDDR2 running at 400 MHz (800MT/s)</li> <li>• Supports Real-Time priority by means of QoS sideband priority signals from the chip to enable different priority levels in the re-ordering mechanism</li> <li>• Page hit/page miss optimizations</li> <li>• Consecutive read/write access optimizations</li> <li>• Supports deep read and write requests queues to enable bank prediction</li> <li>• Drives back the critical word in a read transaction as soon as it is received by the DDR device (doesn't wait until the whole data phase has been completed)</li> <li>• Can track open memory pages</li> <li>• Supports bank interleaving</li> <li>• Special optimization for non-aligned wrap accesses in burst length 8</li> </ul>
AXI interface	<ul style="list-style-type: none"> <li>• AXI bus compliant with glueless interface to PL301 AXI network interconnect</li> <li>• Supports bus transfers of 8,16,32 bits (single accesses and bursts) running at 400 MHz</li> </ul>
DDR calibration and delay-lines	<ul style="list-style-type: none"> <li>• All calibrations can be done automatically by hardware or manually by software</li> <li>• ZQ calibration for external DDR device (in DDR3 through the ZQ calibration command and in LPDDR2 through the MRW command). <ul style="list-style-type: none"> <li>• Can be handled automatically for ZQ Short (periodically) and ZQ Long (at exit from self-refresh).</li> <li>• Can be handled manually at ZQ INIT.</li> </ul> </li> </ul>
DDR general	<ul style="list-style-type: none"> <li>• Configurable timing parameters</li> <li>• Configurable refresh scheme</li> <li>• Supports dynamic voltage, frequency change and low power mode entry through hardware negotiation with the system (req/ack handshake)</li> <li>• Supports automatic self-refresh and power down entry and exit</li> <li>• Supports fast and slow precharge power down in DDR3</li> <li>• Supports various ODT control schemes. <ul style="list-style-type: none"> <li>• Assertion/Deassertion of ODT control per read or write accesses and for active or passive CS</li> </ul> </li> <li>• Supports MRW and MRR commands for LPDDR2.</li> <li>• Software control for moving to derated timing parameters and derated refresh rate according to temperature variation.</li> <li>• Supports various debug and profiling modes</li> </ul>

## 6.3 EIM-PSRAM/NOR Flash controller overview

EIM, which is an external interface module, handles the interface to devices that are external to the chip, including the generation of chip selects, clocks, and control for external peripherals and memory.

It provides asynchronous and synchronous access to devices with an SRAM-like interface.

### 6.3.1 EIM features

- Up to four (software configurable) chip selects for external devices
  - Flexible address decoding; each chip select memory space is determined separately according to the GPR bits in IOMUXC.
  - 128 MByte maximum supported density by default (AUS bit is cleared). When the AUS bit is set, maximum supported density is 32MBytes.
- Selectable write protection for each chip select
- Support for multiplexed address/data bus operation x16 port size
- Programmable data port size for each chip select (x8 and x16 )
- Programmable wait-state generator for each chip select, for write and read accesses separately
- Asynchronous accesses with programmable setup and hold times for control signals
- Supports asynchronous page mode accesses (x16 port size)
- Supports a continuous burst clock that can be used as the reference clock for FPGA
- Independent synchronous memory burst read mode support for NOR-Flash and PSRAM memories (x16 port size)
- Independent synchronous memory burst write mode support for PSRAM and NOR-Flash like memories (CellularRAM™ from Micron, Infineon, and Cypress, OneNAND™ and utRAM™ from Samsung, and COSMORAM™ from Toshiba)
- Supports NAND-Flash devices with NOR-Flash like interface - OneNAND™ (Samsung)
- Independent programmable variable/fix latency support for read and write synchronous (burst) mode
- Support for little endian operation
- ARM AXI slave interface accesses only handled in parallel for single AXI ID transactions
- External interrupt support using the RDY\_INT signal function as an external interrupt pin
- Boot from external device support according to boot signals, using the RDY\_INT signal
  - RDY signal support assertion after reset

- INT signal support assertion after reset for OneNAND™ (Samsung) device
- Supports little endian mode only

### 6.3.2 EIM boot scenarios

EIM allows booting from NOR Flash devices. To select NOR Flash as the boot source, use either the boot mode and configuration GPIO pins or the internal boot-related fuses.

See [Sytem Boot](#) for more information.

### 6.3.3 EIM boot configuration

The following table shows the EIM boot configuration.

**Table 6-2. EIM boot configuration**

EIM_BOOT_CFG bus	EIM affected bits	EIM register
12	NUM16_BYP_GRANT	CS0GCR2
11	DZS[2]	CS0GCR1
10	AUS	CS0GCR1
[9:8]	CSREC[2:1]	CS0GCR1
[7:5]	RWSC[4:2]	CS0GCR1
	WWSC[4:2]	CS0WCR
4	ERRST	WCR
3	RAL	CS0RCR1
	WAL	CS0WCR
2	MUM	CS0GCR1
	OEA[1]	CS0RCR1
[1:0]	DSZ[1:0]	CS0GCR1

### 6.3.4 OneNAND requirements

Because Ready/Busy pin is not in use, OneNAND devices require the following actions:

- Poll the device to see whether it is ready; software performs a read from the device.
- Connect the Ready/Busy signal of the device to any GPIO pin and use it as an interrupt that indicates the on ready state.

# Chapter 7

## System Debug

### 7.1 Overview

This chapter describes the hardware and software debug and application development features and resources of the chip. It discusses the following:

- Core/platform-specific resources
- Resources associated with complex IP blocks
- Chip-wide resources
- Interface to the external debug and development tools

The debug and trace architecture is designed around the following:

- ARM's CoreSight architecture, adapted to i.MX 6SoloLite SoC (for 1x core debug), including a cross-trigger subsystem for cross-domain triggering of debug resources
- JTAG port used to interact with cores under debug by means of SJC, the system JTAG controller port
- DAP, the debug access port that supports the interface to the ARM RealView Debugging tools and other third party tools
- TPIU, a trace port interface unit that efficiently accesses program trace information from the system
- Various chip-wide resources, such as debug features built into the IP blocks and critical signal visibility available through alternate pin functions or observability muxes

### 7.2 Chip and Cortex-A9 Core Platform Debug Architecture

ARM Cortex-A9 Debug architecture is based on CoreSight architecture by ARM Ltd. The CoreSight architecture provides a system wide solution to real-time debug and trace.

The CoreSight architecture is embodied in a set of CoreSight components and compliant processors that form CoreSight systems. Its architecture maintains the traditional requirements of debug and trace:

- To access debug functionality without software interaction;
- To connect to a running system without performing a reset.

Full access to the processor debug capability is available by the ARM Cortex-A9 debug register map through the Advanced Peripheral Bus (APB) slave port. The core includes a Processor Debug Unit allow which stops program execution, examines and alters the processor and coprocessor state, examines and alters the memory and input/output peripheral state and restarts the processor core.

## **7.2.1 Debug Features**

- CoreSight Program Trace Macrocell (PTM): trace generator for the ARM Cortex A9™ core
- Support for a TrustZone-related 3-level debug scheme:
  - Debug in Non-Secure privileged and user, and Secure user
  - Debug in Non-Secure only
- EmbeddedICE-RT logic
  - Support for both monitor-mode and halt-mode debugging:
  - Core run/halt control, debug status/control
  - Breakpoint/watchpoint control
  - Core- and memory-mapped resource examination/modification
- Data communication channel between ARM core and host debugger via JTAG, and the Debug Access Port (DAP) module.
- PMU: Performance Metrics Unit used for system profiling and debug.
- CP15 register for debugging the MMU, I & D L1 cache, and TLB
- PL310 L2 Cache: Provides an event monitoring signal routed to pins for visibility to help in system debug.

The chip includes ARM CoreSight components for multicore debug and trace solutions.

See the following figure for the ARM debug architecture scheme.



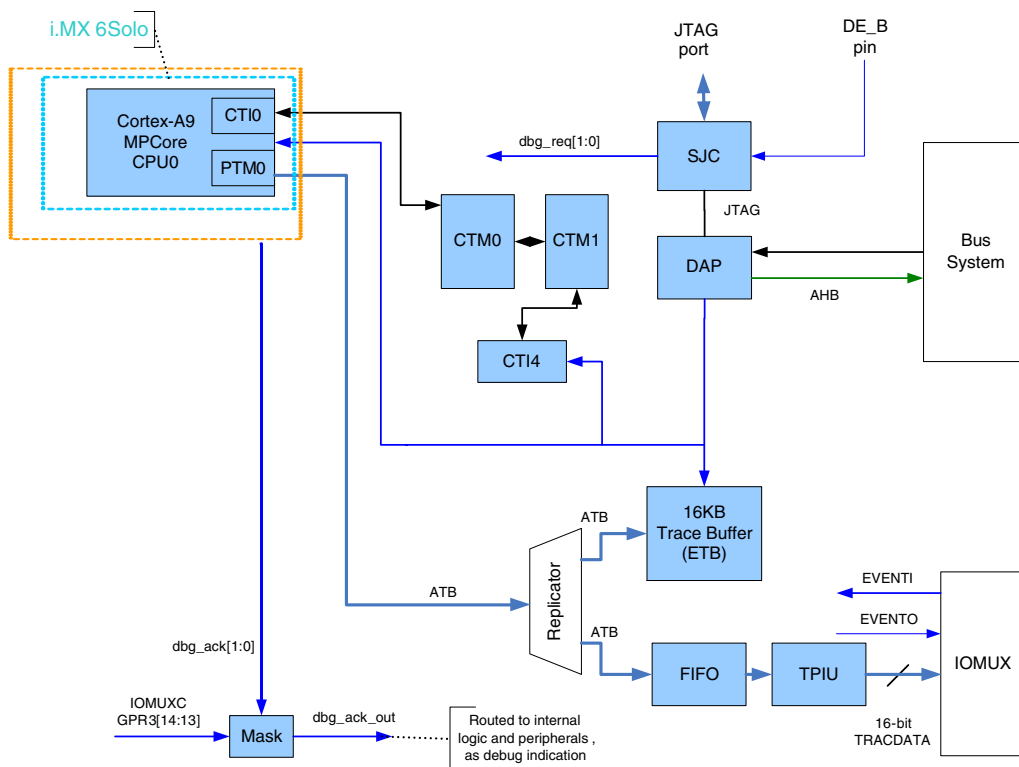


Figure 7-1. i.MX 6SoloLite Cortex-A9 Core Platform Debug Architecture

## 7.2.2 Debug System components

CoreSight components include:

- Embedded Trace Buffer (ETB): RAM array to be used for on-chip capture of trace data output from the PTM
- ATB Replicator to connect the trace data to TPIU (Trace Port Interface) and ETB (Embedded Trace Buffer)
- Cross Triggering logic for event routing, including CTIs and CTMs

Other related IPs and functionality:

- ROMPATCH module to support modification of program/data information in the MCU ROM
- Debug Visibility, which selects critical signals routed to the I/O pads as alternate outputs for external visibility

### 7.2.2.1 AMBA trace bus (ATB)

ATB transfers trace data through the CoreSight infrastructure in a chip. Trace sources are ATB masters and sinks are ATB slaves. The ARM (via PTM) cores are the data generators. Link components such as the Trace Funnel and Replicator provide both master and slave interfaces.

The ATB protocol supports:

- Stalling of trace sources to enable the CoreSight components to funnel and combine sources into a single trace stream.
- Association of the trace data with the generating source using trace source IDs. A CoreSight system can trace up to 111 different items at any one time.
- Capture and transfer of multiple byte bus widths, currently to 32 bits.
- A flushing mechanism to force historic trace to drain from any sources, links, or sinks up to the point that the request was initiated

### 7.2.2.2 ATB replicator

The ATB replicator enables two trace sinks to be wired together and to operate from the same incoming trace stream.

There are no programmable registers. This component is invisible to the user on a particular trace path, from source to sink.

- Incoming ATB Interface—The ATB replicator accepts trace data from the trace source, either directly or through a trace funnel.
- Outgoing ATB Interfaces—The ATB replicator sends identical trace data on outgoing master port interfaces.

### 7.2.2.3 Embedded Cross Triggering

The ECT is a modular component from ARM Limited that supports the interaction and synchronization of multiple triggering events within a chip. The main function of the ECT (CTI and CTM) is to pass debug events from one core to another. For example, the ECT can communicate debug state information from one core to another, so that program execution on both processors can be stopped at the same time if required.

The ECT consists of the following types of modules:

- Cross trigger interfaces (CTI)
- Cross trigger matrix (CTM)

Cross trigger interfaces provide the interface between a component or subsystem and the cross trigger matrix. The system requires a CTI for each subsystem that supports cross triggering. The CTI combines and maps the trigger requests, and broadcasts them to all other interfaces on the ECT as channel events. When the CTI receives a channel event it maps this onto a trigger output. This enables subsystems to cross trigger with each other. The receiving and transmitting of triggers is performed through the trigger interface.

The Cross Trigger Matrix (CTM) combines the trigger request generated from the CTIs and broadcasts them to all CTIs as channel triggers. The CTM controls the distribution of channel events. It provides Channel Interfaces (CIs) to connect to either CTIs or CTMs. This enables multiple CTIs to be linked together. The ECT is composed of three/five CTIs (Cross Trigger Interface) and two CTMs (Cross Trigger Matrix). The ECT is key in the multi-core and multi-IP debug strategy. The outcome is a SWcontrolled debug signal matrix that receives signals from various sources (i.e. cores and peripherals) and propagates/ routes them to the different debug resources of the SoC. Those debug resources can include time stamping capability, profiling capabilities, real-time trace (trace enabled or disabled), triggers, SOC level multiplexing, and debug interrupts.

#### NOTE

As the ECT should only be used during debug sessions, it is off (disabled) by default.

The chip's cross-triggering architecture consists of 2xCTMs and 2<sup>1</sup> CTIs covering three basic groups of functionality as follows:

- CTI 0 - for ARM core

#### 7.2.2.3.1 Cross-Trigger Matrix (CTM)

The CTM (Cross Trigger Matrix) is provided by ARM. A brief description is provided below. It is advised to refer to ARM documentation for more details.

---

1. Depending on specific part: i.MX 6SoloLite.

The CTM is a relatively simple block with no configuration options. There are two CTM instances in i.MX6SoloLite in ARM Platform.

One of them is used to route 1x Core's CTI's, while the second one, is used for additional CoreSight CTI. Each CTI has 4 channel lines, which CTI events are mapped to. The exact mapping is configured in the CTI logic.

#### 7.2.2.3.2 Cross-Trigger Interface (CTI)

The Cross-Trigger Interface (CTI) component is provided by ARM. A brief description of the CTI is provided below.

There are 2 CTIs in the chip's ARM platform. One is of which, dedicated to each Core, while the 2nd is used for various other signals routing.

Each of these CTIs has 8 trigger inputs and 8 trigger outputs that connect to logic in the domain to be debugged or profiled. Each CTI also includes a 4 channel interface to the CTM (4 inputs and 4 outputs).

For more information, see [../././block\\_guide\\_library/arm\\_a9/map\\_arm\\_a9.xml](#).

#### 7.2.2.4 Debug Access Port (DAP)

The DAP enables debug access to the chip modules through APB-AP (the APB access port) and APB-Mux (the APB multiplexer).

AHB-AP provides system access. Debug tools can use JTAG to connect to the chip.

DAP has the following features:

- AMBA 3 Peripheral Bus Multiplexor access through AMBA 3 APB Access Port, providing debug peripheral access through the APB interface.
- External JTAG access using the JTAG Debug Port (JTAG-DP).
- Internal chip module access using:
  - AHB Access Port (AHB-AP)
  - APB Access Port (APB-AP)
  - JTAG Access Port (JTAG-AP)

APB-Mux enables system access to CoreSight components connected to the Debug APB.

The ROM table provides a list of memory locations of CoreSight components connected to the Debug APB. This is visible from both tools and system access and one configures it during system implementation.

External read/write access to the internal interface is provided by JTAG-DP. JTAG-DP provides a standard interface for debug access to the chip through DAP. It interfaces to the DAP internal bus.

Internal access to on-chip buses and other interfaces are provided by the access ports (APs). The available APs are:

- AHB-AP which provides an AHB-Lite master for access to a system AHB bus.
- APB-AP which provides an AMBA 3 APB master for access to the Debug APB that configures all CoreSight components.
- JTAG-AP which provides JTAG access to on-chip components and operates as a JTAG master port to drive JTAG chains throughout the chip.

### 7.2.2.5 CoreSight trace port interface (TPIU)

TPIU is one of the CoreSight trace sink components. It acts as a bridge between the on-chip trace data and a data stream that is then driven out the trace port.

TPIU uses the ATB interface to accept trace data from a trace source, either directly or by using a trace funnel. TPIU has 32 bit port connected to the chip pad.

The APB interface is the programming interface for the TPIU configuration.

The features of the sub-blocks are as follows:

- Formatter—Inserts source ID signals into the data packet stream so that the trace data can be re-associated with the trace source.
- Asynchronous FIFO—Enables trace data to be driven out at a speed that is not dependent on the on-chip bus clock.
- Register Bank—Contains the management, control and status registers for triggers, flushing behavior and external control.
- Trace out—The Trace out block serializes the formatted data before it goes off-chip.
- Pattern Generator—The Pattern Generator unit provides a simple set of defined bit sequences or patterns that can be output over the Trace Port and be detected by the TPA or other associated Trace Capture Device (TCD). The TCD can use these patterns to indicate if it is possible to increase or decrease the trace port clock speed.

The TPIU accepts trace data from a trace source, either direct from a trace source or using a Trace Funnel. The APB interface is the programming interface for the TPIU. The Trace Clock driving the data out to external pins can be obtained from either an on chip or off chip source, selectable via a mux.

The output of the TPIU is connected via external pins (MPS of TRACEDATA in Figure 8-8, which can be in the range of 1–16 bits). The system may utilize double data rate pins to either use a lower clock speed than that of the 32-bit ATB interface, or use fewer than

32 data pins for the output, based on the ability of the technology used. Given the speed of the ATB, 32 data pins with double data rate is recommended, with the external interface running at half the speed of the ATB. The speed of the external interface is from TRACECLKIN, and should be selectable via an on chip or off chip clock. TRACECLK is equal to TRACECLKIN / 2, and is divided in the TPIU to clock trace data at the trace capture unit.

TPIU used to be part of the ARM platform sub blocks in previous versions of i.MX products, placing the TPIU off platform allows future debug trace sources from the chip level to connect to the TPIU by means of a funnel.

For more information, see [Program Trace Macrocell \(PTM\)](#)

## 7.2.3 i.MX6SoloLite-Specific SJC Features

### 7.2.3.1 JTAG Disable Mode

In addition to three different JTAG security modes that are implemented internally in the System JTAG Controller (SJC), there is an option to disable the SJC functionality by e-fuse configuration.

This creates additional JTAG mode "JTAG Disabled" with highest level of JTAG protection. In this mode all JTAG features are disabled. Specifically, the following debug features are disabled in addition to the features that were already disabled in "No Debug" JTAG mode:

- Non-Secure JTAG control registers (PLL configuration, Deterministic Reset, PLL bypass)
- Non-Secure JTAG status registers (Core status)
- Chip Identification Code (IDCODE)

### 7.2.3.2 JTAG ID

**Table 7-1. i.MX JTAG ID**

Device	Silicon revision	JTAG ID (ID CODE)
i.MX 6SoloLite	Rev 1.0	0891_F01Dh <sup>1</sup>

1. In follow-on silicon revisions, the ID value is subject to change by incrementing the first nibble as follows: 1891\_F01Dh for Rev 1.1, 2891\_F01Dh for Rev 1.2 , etc.

## 7.2.4 System JTAG Controller - SJC

The SJC module is the bridge between external development and test instrumentation and the internal JTAG-accessible debug and test resources.

It implements and manages the daisy-chained topology consisting of its own TAP and those of the SDMA, and the ARM Debug Access Port (DAP).

### NOTE

Single Wire Debug (SWD) protocol is not supported.

## 7.2.5 System JTAG controller main features

- IEEE P1149.1, 1149.6 (standard JTAG) interface to off-chip test and development equipment
  - Includes an SJC-only mode for true IEEE P1149.1 compliance, used primarily for board-level implementation of boundary scan.
  - Supports IEEE P1149.6 extensions to the JTAG standard for AC testing of selected I/O signals.
- Debug-related control and status; putting selected cores into reset and/or debug mode and monitoring individual core status signals by means of JTAG
- System status, such as the state of the PLLs (locked or not locked)
- Three levels of security, ranging from no security to no JTAG accessibility to the chip

## 7.2.6 SCJ TAP Port

The SJC supports the following standard JTAG pins:

- TRSTB
- TDI
- TDO
- TCK
- TMS

## 7.2.7 SJC main blocks

- Interface to the outside world via the standard JTAG pins
- Interface to the external Debug\_Event pin

- A master TAP controller which implements the standard JTAG state machine
- Implementation of the mandatory and optional IEEE P1149.1 (JTAG) instructions
  - Mandatory: "EXTEST", "SAMPLE/PRELOAD", and "BYPASS"
  - Optional: "ID\_CODE" (SOC JTAG ID register), "HIGHZ"
- Supports the SDMA's DR-path-only JTAG architecture by implementing the controller portion of its TAP (including "BYPASS" as the default state) within the SJC
- The ExtraDebug registers, which implement a variety of control and status features
  - Three 32-bit insecure general purpose status registers
  - Two 32-bit secure status registers - one predefined, one general purpose.
  - Control and status registers for debug, core, charge pump, and PLL.
- Four levels of fuse-defined security, ranging from no security to no access.

Both predefined and user-defined (SOC integration team) control and status functions are supported by the SJC.

The user-defined functions will be defined and documented by the SOC integration team.

## **7.3 Smart DMA (SDMA) core**

SDMA is a dedicated, programmable DMA engine. It is an integration of a 32-bit RISC core and DMA-specific hardware. It includes ports for the AP domain and a peripheral domain, along with a burst-capable port for direct external memory access.

The SDMA and its integration in the chip is unchanged from previous i.MX chips.

The main SDMA debug features are:

- OnCE - On Chip Emulator, provides the following capabilities:
  - SDMA core control - run/halt/single-step
  - SDMA core register/memory-map access
  - Event detection, watchpoints, and hardware breakpoints
  - Real time buffer and PC trace buffer capability
- Trace buffer
  - Contains information to identify the 32 last changes of flow detected during a program execution
- Context dump
  - Includes information about all the channel dump activity
  - Current contents of SDMA RAM
- ROMPATCH



### 7.3.1 SDMA On Chip Emulation Module (OnCE) Feature Summary

The SDMA debug features are primarily defined by the OnCE portion of its design.

They are summarized as follows:

- **Memory And Register Access** - dedicated logic enables user-access to SDMA memory and register locations. These accesses are supported only when the processor is in debug mode.
- **Event Detection Unit** - watches signals from the data memory bus (DMBus) which is used by the RISC core to access its RAM, ROM, and memory-mapped registers
- **Watchpoints** - one output signal is available to watch event matching conditions at the chip level. Match conditions are defined by programming memory-mapped registers.
- **Hardware Breakpoint** - a counter is decremented after an event detection. A debug request is sent to the SDMA core only when the counter reaches the value of zero. It is possible to program the initial value of the counter or to disable the use of the counter if a debug request must be generated after each event detection.
- **Real Time Buffer** - The Real Time Buffer Register (RTB) is a single 32-bit memory-mapped register which can be accessed as a regular memory location during program execution. It is used to store and retrieve run time information without putting the SDMA in debug mode. Each write to this register causes an event. This register is, in fact, located in the OnCE. Executing through JTAG, a buffer command exports the content of this register through the JTAG port.
- **Core Control (Core Status / Single Stepping)** - Commands are provided to monitor and control processor activity. The commands can halt the core, rerun the core from another address location, and get processor status.
- **Trace Buffer** - a 32x32 buffer which records the last 32 changes of flow during program execution. The buffer stores data in a modulo fashion (i.e. the 33rd instruction change replaces the 1st). Captured trace information is retrieved via reads to the Trace Buffer Register.

#### 7.3.1.1 Other SDMA Debug Functionality

- **Core Trace** - basic core trace capability is available through debug visibility functionality only. PTM trace capability does not exist.
- **ROM Patch** - can be accomplished by manipulating the CHN0ADDR register through JTAG or via the MCU's ability to write to SDMA OnCE registers. This must be done right after reset and before the SDMA core is enabled to begin processing events.
- **Additional debug control/status interaction with the SJC module**

- SJC-controlled Debug Request
- SJC-readable Debug Acknowledge (in debug mode)
- Debug clock control - allows SJC to force clocks on for debug purposes
- Debug core state (SDMA RISC Core State) - 4 bits accessible from the SJC via JTAG

### 7.3.1.2 SDMA ROM Patching

After reset, the SDMA is in its IDLE\_AFTER\_RESET mode. A debug request also puts the SDMA in its DEBUG\_IN\_IDLE\_AFTER\_RESET mode. The new address boot must be stored in CHN0ADDR register (e.g., through the SDMA OnCE via debugger).

The user must then issue the exec\_core <instruction> SDMA OnCE instruction to return to the IDLE\_AFTER\_RESET mode. The very first instructions of the boot code fetches the contents of this register (which is also mapped in the SDMA memory space) and jumps to the given address.

## 7.4 Miscellaneous

### 7.4.1 Clock/Reset/Power

CDBGPWRUPREQ and CDBGPWRUPACK are the handshake signals between the DAP and the clock control module to ensure debug power and clocks are turned on. If the debug components are always powered on, the handshake becomes a mechanism to turn debug clocks on. Similarly, there is a register bit in the CCM which allows internal software to turn debug clocks on as well because the CDBGPWRUPREQ is in the TCLK domain and is inaccessible to software.

The Cortex-A9 and VSP cores can receive resets from the following sources:

- Debug Reset (CDBGIRSTREQ bit within the SWJ-DP CTRL/STAT register of the DAP) in the TCLK domain. This allows the debug tools to reset the debug logic.
- System POR reset

Conversely, the debug system is capable of generating a system reset via a request bit in the MDM-AP control register. This allows the debugger to hold the system in reset.

## 7.5 Supported tools

RealView™ ARM Debugger is supported.

The debugger is connected to the chip from the host by the RealView ICE protocol converter. Other 3rd party tools can be used via the standard JTAG interface, but may need to be adapted for individual IC. It is important to check with tool vendors for specific tool requirements, especially for on-chip IC.



# Chapter 8

## System Boot

### 8.1 Overview

The boot process begins at Power On Reset (POR) where the hardware reset logic forces the ARM core to begin execution starting from the on-chip boot ROM.

Boot ROM code uses the state of the internal register `BOOT_MODE[1:0]` as well as the state of various eFUSES and/or GPIO settings to determine the boot flow behavior of the device.

The main features of the ROM include:

- Support for booting from various boot devices
- Serial downloader support (USB OTG)
- Device configuration data (DCD) and plugin
- Digital signature High Assurance Boot (HAB)
- Wake-up from low power modes
- Plugin image

The boot ROM supports the following boot devices:

- NOR Flash
- OneNAND Flash
- SD/MMC
- Serial (I2C/SPI) NOR Flash and EEPROM

In normal operation, the Boot ROM uses the state of the `BOOT_MODE` register and eFUSES to determine the boot device. For development purposes, eFUSES used to determine the boot device may be overridden by using GPIO pin inputs.

Boot ROM code also allows the downloading of programs to be run on the device. An example is a provisioning program that can make further use of the serial connection to provision a boot device with a new image. Typically the provisioning program is

downloaded to internal RAM and allows the programming of boot devices, such as an SD/MMC Flash. The ROM Serial Downloader uses high speed USB in a non-stream mode connection.

The device configuration data (DCD) feature allows boot ROM code to obtain SOC configuration data from an external Program Image residing on the boot device. As an example, DCD can be used to program the DDR controller for optimal settings improving the boot performance. DCD is restricted to memory areas and peripheral addresses that are considered essential for boot purposes (see [Table 8-29](#)).

A key feature of the boot ROM is the ability to perform a secure boot or High Assurance Boot (HAB). This is supported by the HAB security library which is a subcomponent of the ROM code. HAB uses a combination of hardware and software together with a Public Key Infrastructure (PKI) protocol to protect the system from executing unauthorized programs. Before the HAB allows a user's image to execute, the image must be signed. The signing process is done during the image build process by the private key holder and the signatures are then included as part of the final Program Image. If configured to do so, the ROM verifies the signatures using the public keys included in the Program Image. A secure boot with HAB can be performed on all boot devices supported on the chip in addition to the Serial Downloader. The HAB library in the boot ROM also provides API functions, allowing additional boot chain components (bootloaders) to extend the secure boot chain. The out-of-fab setting for SEC\_CONFIG is the Open configuration in which the ROM/HAB performs image authentication, but all authentication errors are ignored and the image is still allowed to execute.

## 8.2 Boot modes

During reset, the chip checks ARM core ID and Power Gating Controller status register.

On normal boot, the core's behavior is defined by the Boot Mode pins settings as described in [Boot mode pin settings](#). On waking up from low power boot mode, the core skips clock settings. Boot ROM checks that PERSISTENT\_ENTRY0 (see [Persistent Bits](#)) is a pointer to valid address space (OCRAM, DDR or EIM). If PERSISTENT\_ENTRY0 is a pointer to valid range, it starts execution using entry point from PERSISTENT\_ENTRY0 register. If PERSISTENT\_ENTRY0 is a pointer to invalid range, the core performs system reset.

### 8.2.1 Boot mode pin settings

The device has four boot modes (one is reserved for Freescale use). Boot mode is selected based on the binary value stored in the internal BOOT\_MODE register.

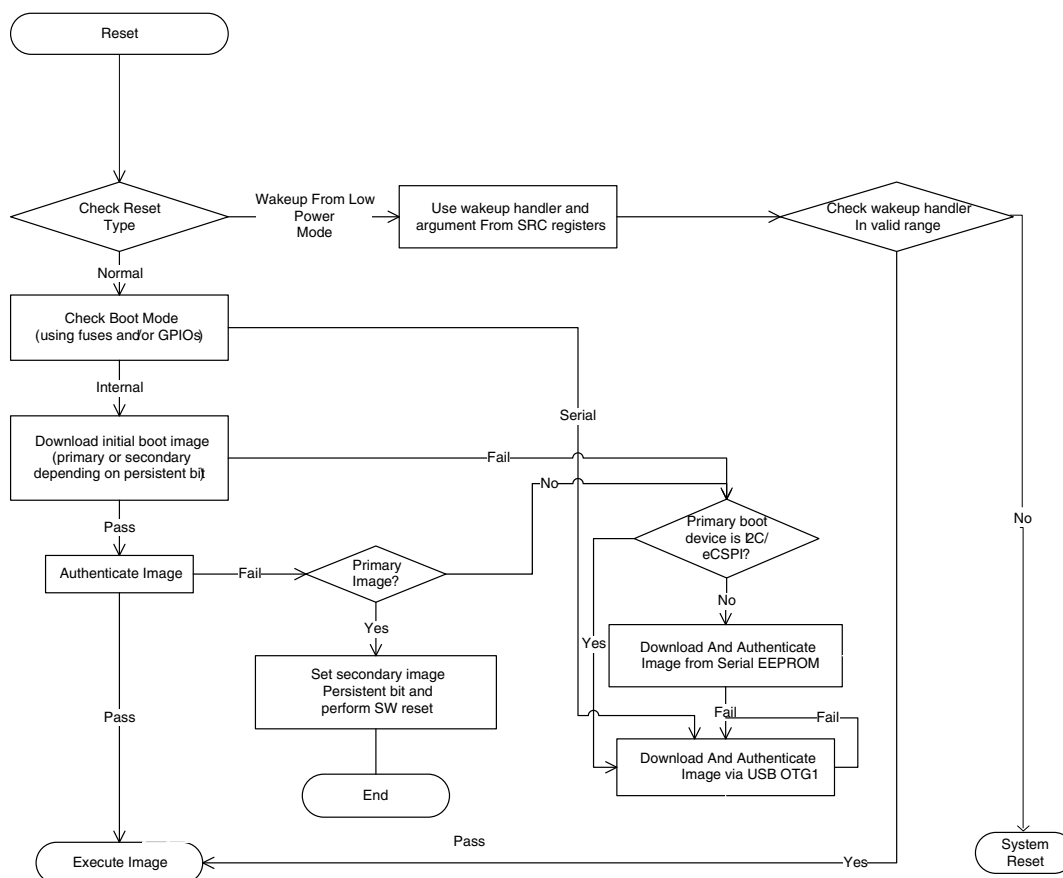
BOOT\_MODE is initialized by sampling the BOOT\_MODE0 and BOOT\_MODE1 inputs on the rising edge of POR\_B. After these inputs are sampled, their subsequent state does not affect the contents of the BOOT\_MODE internal register. The state of the internal BOOT\_MODE register may be read from the BMOD[1:0] field of the SRC Boot Mode Register (SRC\_SBMR2). The available boot modes are: Boot From Fuses, serial boot via USB, and Internal Boot. See the table below for settings.

**Table 8-1. Boot MODE Pin Settings**

BOOT_MODE[1:0]	Boot Type
00	Boot From Fuses
01	Serial Downloader
10	Internal Boot
11	Reserved

## 8.2.2 High level boot sequence

The figure found here shows the high-level boot ROM code flow.



**Figure 8-1. Boot Flow**

## NOTE

For External Interface Module (EIM) boot devices, downloading initial load region to iRAM is skipped. IVT is read from EIM address space (see [Image Vector Table and Boot Data](#)). Copying initial load region and the rest of the program image is done only if the absolute start address of the image is not equal to EIM CS0 start address.

### 8.2.3 Boot From Fuses Mode (BOOT\_MODE[1:0] = 00b)

A value of 00b in the BOOT\_MODE[1:0] register selects the Boot From Fuses mode.

This mode is similar to the Internal Boot mode described in [Internal Boot Mode \(BOOT\\_MODE\[1:0\] = 0b10\)](#) with one difference. In this mode the GPIO boot override pins are ignored. The boot ROM code uses the boot eFUSE settings only. This mode also supports a secure boot using HAB.



If set to Boot From Fuses, the boot flow is controlled by the BT\_FUSE\_SEL eFUSE value. If BT\_FUSE\_SEL = 0, indicating that the boot device (for example, Flash, SD/MMC) has not yet been programmed, the boot flow jumps directly to the Serial Downloader. If BT\_FUSE\_SEL = 1, the normal boot flow is followed, where the ROM attempts to boot from the selected boot device.

The first time a board is used, the default eFUSES may be configured incorrectly for the hardware on the platform. In such a case, the Boot ROM code may try to boot from a device that does not exist. This may cause an electrical/logic violation on some pads. Using Boot From Fuses mode addresses this problem.

The first time the BT\_FUSE\_SEL = 0 eFUSE is encountered, it is not blown (thus setting BT\_FUSE\_SEL). This forces the ROM code to jump directly to the Serial Downloader. This allows a bootloader to be downloaded which can then provision the boot device with a Program Image and blow the BT\_FUSE\_SEL and the other boot configuration eFUSES. After reset, the Boot ROM code determines that BT\_FUSE\_SEL is blown (BT\_FUSE\_SEL = 1) and the ROM code performs internal boot according to the new eFUSE settings. This allows a user to set BOOT\_MODE[1:0]=00b on a production device and burn fuses on the same device (by forcing entry to the Serial Downloader), without changing the value of BOOT\_MODE[1:0] or pullups/pulldowns on the BOOT\_MODE pins.

## 8.2.4 Serial Downloader

The Serial Downloader provides a means to download a Program Image to the chip over USB serial connection.

In this mode the ROM programs WDOG-1 for a 32-second time-out if WDOG\_ENABLE eFuse is 1, and then continuously polls for USB connection. If no activity is found on USB OTG1 and the watchdog timer expires, the ARM core is reset.

### NOTE

The downloaded image must continue to service the watchdog timer to avoid an undesired reset from occurring.

The USB boot flow is shown in the figure below.

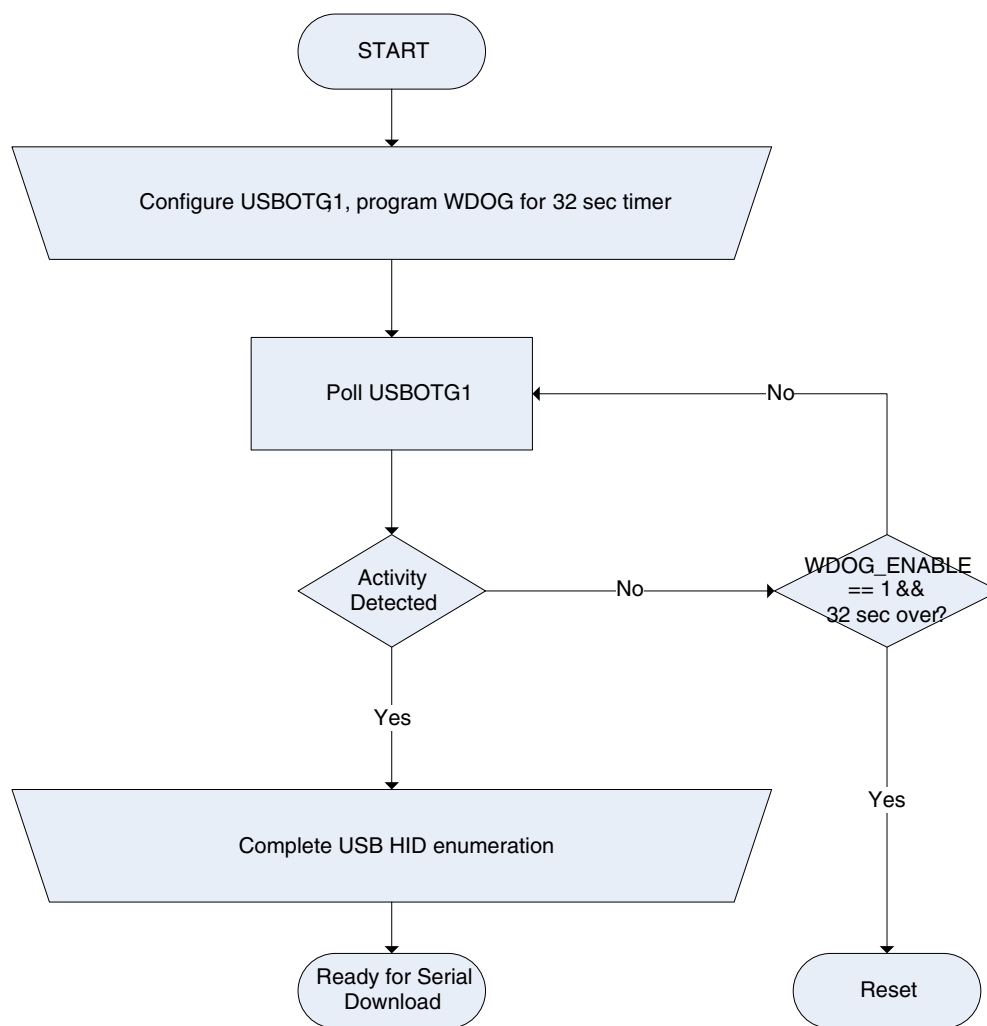


Figure 8-2. USB Boot Flow

### 8.2.5 Internal Boot Mode (BOOT\_MODE[1:0] = 0b10)

A value of 0b10 in the BOOT\_MODE[1:0] register selects the internal boot mode. In this mode, the processor continues to execute boot code from the internal boot ROM.

The boot code performs hardware initialization, loads the Program Image from the chosen boot device, performs image validation using the HAB library (see [Boot security settings](#)), and then jumps to an address derived from the Program Image. If any error occurs during internal boot, the boot code jumps to the Serial Downloader (see [Serial Downloader](#)). A secure boot using the HAB is possible in all the three boot modes.

When set to internal boot, the boot flow may be controlled by a combination of eFUSE settings with an option of overriding the fuse settings using General Purpose I/O (GPIO) pins. The GPIO Boot Select FUSE (BT\_FUSE\_SEL) determines whether the ROM uses GPIO pins for a select number of configuration parameters or eFUSES in this mode. See [Table 8-4](#) for more details.

- If BT\_FUSE\_SEL = 1, all boot options are controlled by the eFUSES described in [Table 8-2](#).
- If BT\_FUSE\_SEL = 0, specific boot configuration parameters may be set using GPIO pins rather than eFUSES. The fuses that can be overridden when in this mode are indicated in the GPIO column of [Table 8-2](#). [Table 8-3](#) provides the details on the GPIO pins.

The use of GPIO overrides is intended for development since these pads are used for other purposes in deployed products. Freescale recommends controlling the boot configuration by eFUSES in deployed products and reserving the use of the GPIO mode for development and testing purposes only.

## 8.2.6 Boot security settings

Internal boot modes use one of three security configurations:

- **Closed:** This level is intended for use with shipping secure products. All HAB functions are executed and security hardware is initialized (the Security Controller, or SNVS, enters Secure state), DCD is processed if present, and the program image is authenticated by HAB prior to its execution. All detected errors will be logged, and the boot flow aborted with control passing to the serial downloader. At this level, execution does not leave the internal ROM unless the target executable image has been authenticated.
- **Open:** This level is intended for use in non-secure products or during the development phases of a secure product. All HAB functions are executed as for a closed device. Security hardware is initialized (except the SNVS is left in Non-Secure state), DCD is processed if present, and the program image is authenticated by HAB prior to its execution. All detected errors will be logged, but have no influence on the boot flow, which continues as if the errors did not occur. This configuration is useful for secure product development, since the Program Image will run even if the authentication data is missing or incorrect, and the error log can be examined to determine the cause of authentication failure.
- **Field Return:** This level is intended for parts returned from shipped products.

**NOTE**

If the DIR\_BT\_DIS eFuse is not blown, authentication may be bypassed. In this case the system is not secure.

## 8.3 Device Configuration

This section describes the external inputs that control the behavior of the Boot ROM code.

This includes boot device selection (SPI, EIM, NOR, SD, MMC, etc.), boot device configuration (SD bus width, speed, etc), and so on. In general, the source for this configuration comes from eFUSES embedded inside the chip. However, certain configuration parameters can be sourced from GPIO pins allowing further flexibility during the development process.

### 8.3.1 Boot eFUSE Descriptions

The table below is a comprehensive list of the configuration parameters that the ROM uses.

**Table 8-2. Boot eFUSE Descriptions**

Fuse	Configuration	Definition	GPIO <sup>1</sup>	Shipped Value	Settings <sup>2</sup>
DIR_BT_DIS	OEM	Disables Freescale reserved modes. Must be set for secure boot.	NA	0	0 Reserved Freescale modes enabled 1 Reserved Freescale modes disabled
BT_FUSE_SEL	OEM	In internal Boot mode BOOT_MODE[1:0] = 10, the BT_FUSE_SEL fuse determines whether the boot settings indicated by a Yes in the GPIO column are controlled by GPIO pins or eFUSE settings in the On-Chip OTP Controller (OCOTP).  In Boot From Fuse mode BOOT_MODE[1:0] = 00, BT_FUSE_SEL fuse indicates whether bit configuration eFuses have been programmed.	NA	0	If BOOT_MODE[1:0] = 0b10 0 Bits of SBMR are overridden by GPIO pins. 1 Specific bits of SBMR are controlled by eFUSE settings.  If BOOT_MODE[1:0] = 0b00 0 BOOT configuration eFuses are not yet programmed. Boot flow jumps to serial downloader. 1 BOOT configuration eFuses have been programmed. Regular boot flow is performed.

*Table continues on the next page...*

**Table 8-2. Boot eFUSE Descriptions  
(continued)**

Fuse	Config uratio n	Definition	GPIO <sup>1</sup>	Shipped Value	Settings <sup>2</sup>
SEC_CONFIG[1:0]	SEC_CONFIG[0] - Freescale SEC_CONFIG[1] - OEM	Security Configuration as defined in <a href="#">Boot security settings</a>	NA	01	00 Reserved 01 Open (allows any program image, even if authentication fails) 1x Closed (Program image executes only if authenticated)
FIELD_RETURN	OEM	Enables Freescale reserved modes			0 - Freescale reserved modes are enabled/disabled based on DIR_BT_DIS value 1 - Freescale reserved modes are enabled
SRK_HASH[255:0]	OEM	256-bit hash value of super root key (SRK_HASH)	NA	0	Settings vary - used by HAB
DIE-X-COORDINATE[7:0] DIE-Y-COORDINATE[7:0] WAFER_NO[4:0] LOT_NO_ENC[42:40] LOT_NO_ENC[39:32] LOT_NO_ENC[31:24] LOT_NO_ENC[23:16] LOT_NO_ENC[15:8] LOT_NO_ENC[7:0]	Freescale	Device Unique ID, 64-bit UID.	NA	Unique ID	Settings vary - used by HAB
BT_MMU_DISABLE	OEM	MMU/L1 D Cache/PL310 disable bit used by boot ROM for fast HAB processing	No	0	0 - MMU/L1 D Cache/PL310 is enabled by ROM during the boot 1 - MMU/L1 D Cache/PL310 is disabled by ROM during the boot
L1 I-Cache DISABLE	OEM	L1 I Cache disable bit used by boot during entire execution	No	0	0 - L1 I Cache is enabled by ROM during the boot 1 - L1 I Cache is disabled by ROM during the boot
BT_FREQ (BOOT_CFG2[2])	OEM	Frequency Selection	Yes	0	0 - ARM - 792MHz, DDR - 532MHz, AXI - 264MHz 1 - ARM - 396MHz, DDR - 352MHz, AXI - 176MHz
BOOT_CFG1[7:0]	OEM	Boot Configuration1	Yes	0	Specific to selected boot mode
BOOT_CFG2[7:0]	OEM	Boot Configuration2	Yes	0	Specific to selected boot mode
BOOT_CFG4[6:0]	OEM	Boot Configuration4		0	Specific to selected boot mode

Table continues on the next page...

**Table 8-2. Boot eFUSE Descriptions  
(continued)**

Fuse	Configuration	Definition	GPIO <sup>1</sup>	Shipped Value	Settings <sup>2</sup>
BOOT_CFG4[7]	OEM	Infinite Loop Enable at start of boot ROM. Used for debugging purposes. Ignored if DIR_BT_DIS is 1 and FIELD_RETURN is 0.	Yes	0	0 - Disabled 1 - Enabled
LPB_BOOT	OEM	USB Low Power Boot	No	0	00 - LPB Disable 01 - 1 GPIO (default frequencies) 10 - Divide by 2 11 - Divide by 4
BT_LPB_POLARITY	OEM	USB Low Power Boot GPIO polarity	No	0	0 - low on GPIO pad indicates low power condition 1 - high on GPIO pad indicates low power condition
WDOG_ENABLE	OEM	Watchdog reset counter enable	No	0	0 - watchdog reset counter is disabled during serial downloader 1 - watchdog reset counter is enabled during serial downloader
MMC_DLL_DLY[6:0]	OEM	uSDHC Delay Line settings	No	0	uSDHC Delay Line settings
SRK_REVOKE[2:0]	OEM	SRK revocation mask	No	0	SRK revocation mask
DISABLE_SDMMC_MFG	OEM	Disable SDMMC manufacture mode	No	0	0: enable SD/MMC MFG mode 1: disable SD/MMC MFG mode
PAD_SETTINGS	OEM	Override values for SD/MMC and NAND boot modes	No	0	Override the following IO PAD settings: PAD_SETTINGS[0] - Slew Rate PAD_SETTINGS[3:1] Drive Strength PAD_SETTINGS[5:4] - Speed Settings .
USE_L2_CACHE_AS_OCRAM (BOOT_CFG1[7])	OEM	L2 cache memory to be configured as OCRAM	Yes	0	0: L2 cache will be used as cache 1: L2 cache will be used as OCRAM <sup>3</sup>
OVERRIDE_HYS_SDMMC_PADS	OEM	Overrides HYS bit for SD pads	No	0	Override the IO PAD setting HYS to 1 for SD pads
eMMC_4.4_RESET_TO_PRE-IDLE_STATE	OEM	ROM reset the boot device in pre-idle state using eMMC 4.4 feature, CMD0 with argument value 0xf0f0f0	No	0	Applicable for booting from eMMC 4.4 spec or greater version devices. The fuse should not be blown for eMMC 4.3 or lesser spec version devices.

1. Setting can be overridden by GPIO settings when BT\_FUSE\_SEL fuse is intact. See [Table 1](#) for corresponding GPIO pin.
2. 0 = intact fuse and 1= blown fuse
3. L2 Cache will be locked after boot to prevent the leaking of cache contents.

## 8.3.2 GPIO Boot Overrides

The table below provides a list of GPIO boot overrides.

**Table 8-3. GPIO Override Contact Assignments**

Package Pin	Direction on reset	eFuse
BOOT_MODE1	Input	Boot Mode Selection
BOOT_MODE0	Input	
LCD_DAT0	Input	BOOT_CFG1[0]
LCD_DAT1	Input	BOOT_CFG1[1]
LCD_DAT2	Input	BOOT_CFG1[2]
LCD_DAT3	Input	BOOT_CFG1[3]
LCD_DAT4	Input	BOOT_CFG1[4]
LCD_DAT5	Input	BOOT_CFG1[5]
LCD_DAT6	Input	BOOT_CFG1[6]
LCD_DAT7	Input	BOOT_CFG1[7]
LCD_DAT8	Input	BOOT_CFG2[0]
LCD_DAT9	Input	BOOT_CFG2[1]
LCD_DAT10	Input	BOOT_CFG2[2]
LCD_DAT11	Input	BOOT_CFG2[3]
LCD_DAT12	Input	BOOT_CFG2[4]
LCD_DAT13	Input	BOOT_CFG2[5]
LCD_DAT14	Input	BOOT_CFG2[6]
LCD_DAT15	Input	BOOT_CFG2[7]
LCD_DAT16	Input	BOOT_CFG4[0]
LCD_DAT17	Input	BOOT_CFG4[1]
LCD_DAT18	Input	BOOT_CFG4[2]
LCD_DAT19	Input	BOOT_CFG4[3]
LCD_DAT20	Input	BOOT_CFG4[4]
LCD_DAT21	Input	BOOT_CFG4[5]
LCD_DAT22	Input	BOOT_CFG4[6]
LCD_DAT23	Input	BOOT_CFG4[7]

The input pins provided are sampled at boot, and can be used to override corresponding eFUSE values, depending on the setting of the BT\_FUSE\_SEL fuse. Table below describes boot options control sampling in different boot modes.

**Table 8-4. Boot Options Control Selection**

BOOT_MODE[1:0]	BT_FUSE_SEL Value	Boot Options Controlled By
00	0	eFUSEs
	1	

*Table continues on the next page...*

**Table 8-4. Boot Options Control Selection  
(continued)**

BOOT_MODE[1:0]	BT_FUSE_SEL Value	Boot Options Controlled By
01	0	GPIO pins
	1	eFUSES
10	0	GPIO pins
	1	eFUSES

### 8.3.3 Device Configuration Data

DCD is configuration information contained in a Program Image, external to the ROM, that the ROM interprets to configure various on-chip peripherals. See [Device Configuration Data \(DCD\)](#) for more details on Device Configuration Data.

## 8.4 Device Initialization

This section describes the details on the ROM and provides initialization details.

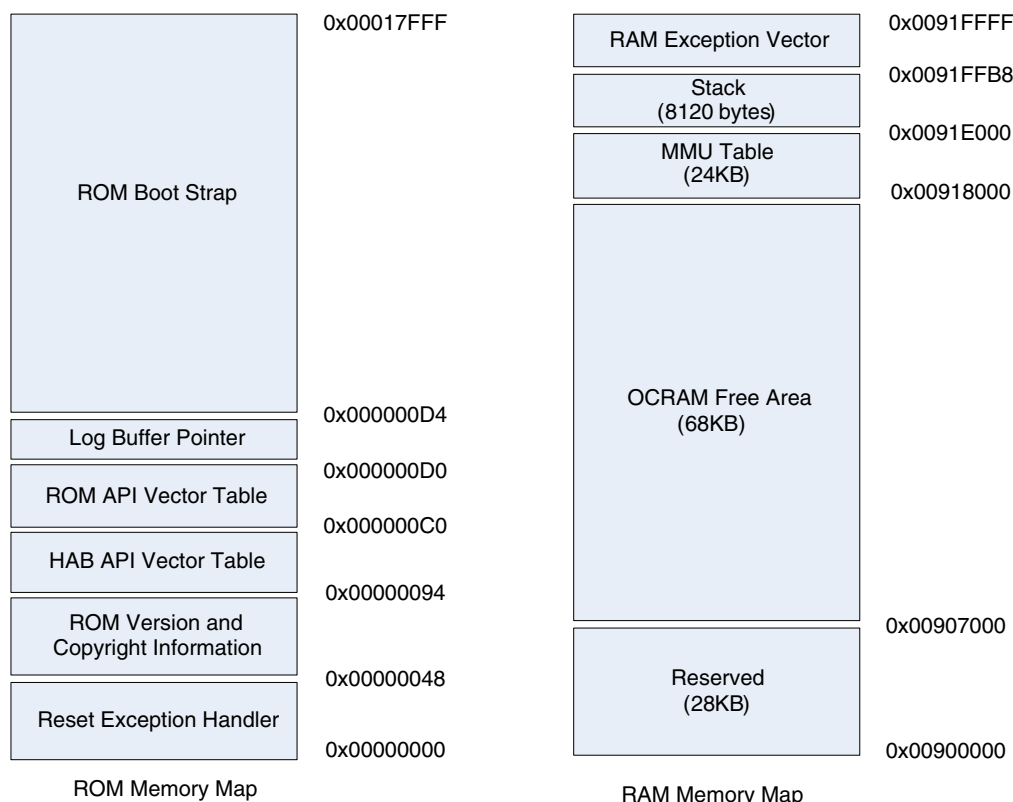
This includes details on:

- The ROM Memory Map
- The RAM Memory Map
- On-chip blocks that the ROM should make use of or change POR register default values
- Clock initialization
- Enabling the MMU/L2 cache when performing a secure boot (SEC\_CONFIG = Closed)
- Exception handling and interrupt handling

### 8.4.1 Internal ROM /RAM memory map

[Figure 8-3](#) shows the iROM memory map for the i.MX 6SoloLite.





**Figure 8-3. Internal Rom and Ram memory map for i.MX 6SoloLite**

### NOTE

The entire OCRM region can be used freely post boot.

## 8.4.2 Boot Block Activation

The boot ROM affects a number of different hardware blocks which are activated and play a vital role in the boot flow.

The ROM configures and uses the following blocks (listed in alphabetical order) during the boot process. Note that the blocks actually used depend on the boot mode and boot device selection:

- CCM - Clock Control Module
- ECSPI - Enhanced Configurable Serial Peripheral Interface
- EIM - External Interface Module. Used for NOR and OneNAND devices
- I2C - I2C Controller
- OCOTP\_CTRL - On-Chip OTP Controller. The OCOTP contains the eFUSES.

- IOMUXC - I/O Multiplexer Control allows GPIO use to override eFUSE boot settings
- DCP - Data CoProcessor Module for Hash acceleration
- SNVS - Secure Non-Volatile Storage
- SRC - System Reset Controller
- USB - Used for serial download of a boot device provisioning program
- USDHC - Ultra Secure Digital Host Controller
- WDOG-1 - Watchdog Timer

### 8.4.3 Clocks at Boot Time

The table below shows the various clocks and their sources used by ROM.

**Table 8-5. Normal Frequency Clocks Configuration**

Clock	CCM signal	Source	Frequency(MHz) BT_FREQ=0	Frequency(MHz) BT_FREQ=1
System PLL	pll1_sw_clk		792	792
528MHz PLL	pll2_sw_clk		528	528
480MHz PLL	pll3_sw_clk		480	480
ARM core clock	arm_clk_root	System PLL	792	396
EIM	emi_slow_clk_root	528MHz PLL	132	132
AHB	ahb_clk_root	528MHz PLL/PFD307	132	76.6
IPG	ipg_clk_root	528MHz PLL/PFD307	66	38.3
AXI_A	axi_a	528MHz PLL/PFD307	396	306
AXI_B	axi_b	528MHz PLL/PFD352	264	153.3
USB	usb0h3_clk_root	480MHz PLL	60	60
USDHC	usdhc1_clk_root usdhc2_clk_root usdhc3_clk_root usdhc4_clk_root	PFD400	198	198
ECSPI	ecspi_clk_root	480MHz PLL	60	60
I2C	per_clk_root	528MHz PLL	66	38.3

Following reset, each ARM core has access to all peripherals. The ROM code will disable the clocks listed in the following table, except for the boot devices listed in the second column.

**Table 8-6. List Of Disabled Clocks**

Clock Name	Enabled For Boot Device
CCGR0_APBHDMA	
CCGR1_ECSP11	ECSP11
CCGR1_ECSP12	ECSP12
CCGR1_ECSP13	ECSP13
CCGR1_ECSP14	ECSP14
CCGR1_FEC	
CCGR1_EPIT1	
CCGR1_EPIT2	
CCGR2_I2C1_SERIAL	I2C1
CCGR2_I2C2_SERIAL	I2C2
CCGR2_I2C3_SERIAL	I2C3
CCGR3_CSI_CORE	
CCGR3_PXP_AXI	
CCGR3_EPDC_AXI	
CCGR3_LCDIF_AXI	
CCGR3_LCDIF_PIX	
CCGR3_EPDC_PIX	
CCGR3_OPENVGAXICLK	
CCGR4_PWM1	
CCGR4_PWM2	
CCGR4_PWM3	
CCGR4_PWM4	
CCGR5_SDMA	
CCGR5_SPDIF	
CCGR5_SSI1	
CCGR5_SSI2	
CCGR5_SSI3	
CCGR5_UART	
CCGR5_UART_SERIAL	
CCGR6_USBOH3	USB
CCGR6_USDHC1	USDHC1
CCGR6_USDHC2	USDHC2
CCGR6_USDHC3	USDHC3
CCGR6_USDHC4	USDHC4
CCGR6_EMI_SLOW	NOR, OneNAND

## 8.4.4 Enabling MMU and Caches

The boot ROM includes a feature of enabling the Memory Management Unit (MMU) and caches to improve boot speed when performing a secure boot with SEC\_CONFIG=Closed ( [High Assurance Boot \(HAB\)](#)). L1 instruction cache is enabled at the start of image download. L1 data cache, L2 cache and MMU are enabled during image authentication. Once HAB authentication completes the ROM disables the L2 cache and MMU.

L1 Instruction cache is controlled by L1 I-Cache eFuse. By default the L1 I-Cache is enabled.

Enabling the MMU when booting non-securely with SEC\_CONFIG=Open, and setting the CSF pointer in the Image Vector Table to NULL, has no impact on the boot performance. With this configuration it is recommended to blow BT\_MMU\_DISABLE fuse.

## 8.4.5 Exception Handling

The exception vectors located at the start of ROM are used to map all the ARM exceptions (except the reset exception) to a duplicate exception vector table in internal RAM.

During the boot phase of CPU0, the RAM vectors point to the serial downloader in ROM.

During the boot phase of a secondary CPU, the internal RAM vectors point to a function that sets the error status registers (see [Persistent Bits](#)), sends a wakeup error interrupt and performs the Wait For Interrupt instruction. The interrupt service routine of primary CPU must reconfigure the system and reset the secondary CPU.

After boot the program image can overwrite the vectors as required. The code shown below is used to map the ROM exception vector table to the duplicate one in RAM.

### Mapping ROM Exception Vector Table

```
;; Define linker area for ROM exception vector table
AREA IROM_VECTORS, CODE, READONLY
LDR    PC, Reset_Addr
LDR    PC, Undefined_Addr
LDR    PC, SWI_Addr
LDR    PC, Prefetch_Addr
LDR    PC, Abort_Addr
NOP                                ; Reserved vector
LDR    PC, IRQ_Addr
LDR    PC, FIQ_Addr
LDR    PC, SW_monitor_addr
;; Define exception vector table
Reset_Addr    DCD    start_address
Undefined_Addr DCD    iRAM_Undefined_Handler
```

```

SWI_Addr      DCD      iRAM_SWI_Handler
Prefetch_Addr DCD      iRAM_Prefetch_Handler
Abort_Addr    DCD      iRAM_Abort_Handler
              DCD      0 ; Reserved vector
IRQ_Addr      DCD      iRAM_IRQ_Handler
FIQ_Addr      DCD      iRAM_FIQ_Handler
SW_monitor_add DCD      SW monitor exception
start_address DCD start ;reset handler vector

```

## 8.4.6 Interrupt Handling During Boot

No special interrupt handling routines are required during the boot process. Interrupts are disabled during boot ROM execution and may be enabled in a later boot stage.

## 8.4.7 Persistent Bits

Some modes of boot ROM require registers that keep their values after warm reset. SRC General Purpose registers are used for this purpose.

See the table below for persistent bits list and description.

**Table 8-7. Persistent Bits**

Bit Name	Bit Location	Description
PERSIST_SECONDARY_BOOT	SRC_GPR10[30]	This bit identifies which image must be used - primary and secondary. Used only for boot modes that support redundant boot.
PERSISTENT_ENTRY0[31:0]	SRC_GPR1[31:0]	Holds entry function for CPU0 for waking-up from low power mode.
PERSISTENT_ARG0[31:0]	SRC_GPR2[31:0]	Holds argument of entry function for CPU0 for waking-up from low power mode.

## 8.5 Boot Devices (Internal Boot)

The Chip supports the following boot Flash devices:

- NOR Flash with External Interface Module (EIM), located on CS0, 16-bits and 32-bit bus width
- OneNAND Flash with EIM interface, located on CS0, 16-bits bus width
- SD/MMC/eSD/SDXC/eMMC4.4 via USDHC interface, supporting high capacity cards
- EEPROM boot via SPI (serial flash) and I2C (via ECSPI and I2C blocks respectively)

The selection of external boot device type is controlled by BOOT\_CFG1[7:4] eFUSES. See the table below for more details.

**Table 8-8. Boot Device Selection**

BOOT_CFG1[7:4]	Boot Device
0000	NOR/OneNAND (EIM)
0001	Reserved
0011	Serial ROM (I2C/SPI)
010x	SD/eSD/SDXC
011x	MMC/eMMC

### 8.5.1 NOR Flash/OneNAND using EIM Interface

The External Interface Module (EIM) works in the asynchronous mode, and supports either muxed, Address/Data, or non-muxed schemes based on fuse settings.

**Table 8-9. EIM Boot eFUSE Descriptions**

Fuse	Config	Definition	GPIO <sup>1</sup>	Shipped Value	Settings
BOOT_CFG1[7:4]	OEM	Boot Device Selection	Yes	0000	0000 - Boot from EIM Interface
BOOT_CFG1[3]	OEM	NOR/OneNAND Selection	Yes	0	0 - NOR 1 - OneNAND
BOOT_CFG2[7:6]	OEM	Muxing Scheme	Yes	00	00 - Muxed, 16-bit data (low half) interface 01 - Reserved 10 - Not muxed, 16-bit data (low half) interface 11 - Reserved
BOOT_CFG2[5:4]	OEM	OneNAND Page Size	Yes	00	00 - 1K 01 - 2K 10 - 4K 11 - Reserved

1. Setting can be overridden by GPIO settings when BT\_FUSE\_SEL fuse is intact. See [Table 1](#) for corresponding GPIO pin.

#### 8.5.1.1 NOR Flash Boot Operation

Booting from the NOR Flash is supported via EIM interface. The ROM reads Image Vector Table and Boot Data structures to determine if the image can be executed directly from EIM address space or should be copied to other memory.

The start field of Boot Data Structure specifies the final location of the image (see [Image Vector Table and Boot Data](#)).

### 8.5.1.2 OneNAND Flash Boot Operation

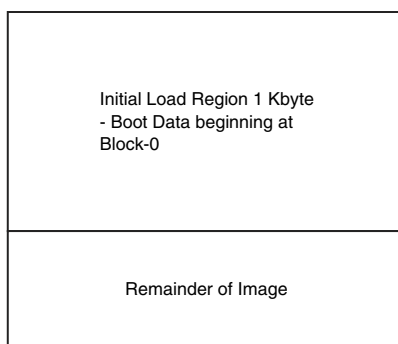
At system power-up, the OneNAND device automatically copies an Initial Load Region of 1 Kbyte from the start of the flash array (sector 0 and sector 1, page 0, block 0) to its Boot RAM (OneNAND's internal RAM).

#### NOTE

The OneNAND boot RAM memory containing the Initial 1K Load Region must contain the IVT, DCD and the Boot Data structures.

Next, the ROM processes the DCD and then proceeds to copy the Program Image contents to the application destination pointer (located in the start entry of Boot Data (see [Image Vector Table and Boot Data](#)). The ROM determines the size of the Program Image by the length specified by size entry in Boot Data structure (see [Image Vector Table and Boot Data](#)). A failure loading data from the OneNAND device for any reason forces the Chip to enter the Serial Downloader, otherwise the booting from the OneNAND device continues.

The figure below illustrates the layout of the Program Image on a OneNAND boot device.



**Figure 8-4. Program Image Layout on a OneNAND Flash Device**

Prior to accessing the OneNAND device, the Chip waits approximately 500  $\mu$ s after Power On Reset. This delay is required for the OneNAND device to become ready. After this initial 500  $\mu$ s delay it can take an addition 70  $\mu$ s for the OneNAND device to load the Initial Load Region of 1 Kbyte into its boot RAM. The Chip polls the OneNAND device Interrupt Status Register to confirm that the first 1 Kbytes has been loaded to the OneNAND boot RAM before continuing with the boot flow.

### 8.5.1.3 IOMUX Configuration for EIM Devices

The EIM interface uses dedicated contacts on the IC.

The contacts assigned to the data signals used by EIM are shown in the table below.

**Table 8-10. EIM IOMUX Pin Configuration**

Signal	Muxed	Not muxed
DATA0	KEY_COL0.alt3	LCD_DAT6.alt3
DATA1	KEY_ROW0.alt3	LCD_DAT7.alt3
DATA2	KEY_COL1.alt3	LCD_DAT8.alt3
DATA3	KEY_ROW1.alt3	LCD_DAT9.alt3
DATA4	KEY_COL2.alt3	LCD_DAT10.alt3
DATA5	KEY_ROW2.alt3	LCD_DAT11.alt3
DATA6	KEY_COL3.alt3	LCD_DAT12.alt3
DATA7	KEY_ROW3.alt3	LCD_DAT13.alt3
DATA8	KEY_COL4.alt3	LCD_DAT14.alt3
DATA9	KEY_ROW4.alt3	LCD_DAT15.alt3
DATA10	KEY_COL5.alt3	LCD_DAT16.alt3
DATA11	KEY_ROW5.alt3	LCD_DAT17.alt3
DATA12	KEY_COL6.alt3	LCD_DAT18.alt3
DATA13	KEY_ROW6.alt3	LCD_DAT19.alt3
DATA14	KEY_COL7.alt3	LCD_DAT20.alt3
DATA15	KEY_ROW7.alt3	LCD_DAT21.alt3
ADDR0	KEY_COL0.alt3	
ADDR1	KEY_ROW0.alt3	
ADDR2	KEY_COL1.alt3	
ADDR3	KEY_ROW1.alt3	
ADDR4	KEY_COL2.alt3	
ADDR5	KEY_ROW2.alt3	
ADDR6	KEY_COL3.alt3	
ADDR7	KEY_ROW3.alt3	
ADDR8	KEY_COL4.alt3	
ADDR9	KEY_ROW4.alt3	
ADDR10	KEY_COL5.alt3	
ADDR11	KEY_ROW5.alt3	
ADDR12	KEY_COL6.alt3	
ADDR13	KEY_ROW6.alt3	
ADDR14	KEY_COL7.alt3	
ADDR15	KEY_ROW7.alt3	
ADDR16	EPDC_D8.alt3	

*Table continues on the next page...*



**Table 8-10. EIM IOMUX Pin Configuration (continued)**

Signal	Muxed	Not muxed
ADDR17		EPDC_D9.alt3
ADDR18		EPDC_D10.alt3
ADDR19		EPDC_D11.alt3
ADDR20		EPDC_D12.alt3
ADDR21		EPDC_D13.alt3
ADDR22		EPDC_D14.alt3
ADDR23		EPDC_D15.alt3
ADDR24		EPDC_VCOM0.alt3
ADDR25		EPDC_VCOM1.alt3
ADDR26		EPDC_BDR0.alt3
CS0		EPDC_PWRCTRL2.alt3
OE		EPDC_PWRCTRL1.alt3
RW		EPDC_PWRCTRL0.alt3
LBA		EPDC_SDCE1.alt3

## 8.5.2 Expansion Device

The ROM supports booting from MMC/eMMC and SD/eSD compliant devices.

### 8.5.2.1 Expansion Device eFUSE Configuration

SD/MMC/eSD/eMMC/SDXC boot can be performed using either USDHC-1, USDHC-2, USDHC-3, or USDHC-4 ports, based on setting of the BOOT\_CFG2[4:3] (Port Select) fuse or it's associated GPIO input value at boot. All USDHC ports support eMMC4.3 and eMMC4.4 fast boot.

See the table below for details.

**Table 8-11. USDHC Boot eFUSE Descriptions**

Fuse	Config	Definition	GPIO <sup>1</sup>	Shipped Value	Settings
BOOT_CFG1[7:6]	OEM	Boot Device Selection	Yes	00	01 - Boot from USDHC Interfaces
BOOT_CFG1[5]	OEM	SD/MMC Selection	Yes	0	0 - SD/eSD/SDXC 1 - MMC/eMMC
BOOT_CFG1[4]	OEM	Fast Boot Support	Yes	0	0 - Normal Boot 1 - Fast Boot

*Table continues on the next page...*

**Table 8-11. USDHC Boot eFUSE Descriptions  
(continued)**

Fuse	Config	Definition	GPIO <sup>1</sup>	Shipped Value	Settings
BOOT_CFG1[3:2]	OEM	SD/MMC Speed Mode	Yes	00	MMC 0x - High Speed Mode 1x - Normal Speed Mode SD 0x - High/Normal 10 - SDR50 11 - SDR104
BOOT_CFG1[1]	OEM	SD Power Cycle Enable/ eMMC Reset Enable	Yes	0	MMC 0 - eMMC reset disabled 1 - eMMC reset enabled via SD_RST pad (on USDHC3 and USDHC4 only) SD 0 - No power cycle 1 - Power cycle enabled via SD_RST pad (on USDHC3 and USDHC4 only)
BOOT_CFG1[0]	OEM	SD Loopback Clock Source Sel(for SDR50 and SDR104 only)	Yes	00	0 - through SD pad 1 - direct
BOOT_CFG2[7:5]	OEM	Bus Width/SD Calibration Step	Yes	000	SD/eSD/SDXC (BOOT_CFG1[5]=0) Bus Width xx0 - 1-bit xx1 - 4-bit SD Calibration Step 00x - 1 delay cells 01x - 1 delay cells 10x - 2 delay cells 11x - 3 delay cells MMC/eMMC (BOOT_CFG1[5]=1) 000 - 1-bit 001 - 4-bit 010 - 8-bit 101 - 4-bit DDR (MMC 4.4) 110 - 8-bit DDR (MMC 4.4) Else - reserved.

*Table continues on the next page...*

**Table 8-11. USDHC Boot eFUSE Descriptions  
(continued)**

Fuse	Config	Definition	GPIO <sup>1</sup>	Shipped Value	Settings
BOOT_CFG2[4:3]	OEM	Port Select	Yes	00	00 - USDHC-1 01 - USDHC-2 10 - USDHC-3 11 - USDHC-4
BOOT_CFG2[2]	OEM	DLL Override (eMMC Only)	Yes	0	0 - Boot ROM default. 1 - Apply value per fuse field MMC_DLL_DLY[6:0]
BOOT_CFG2[1]	OEM	Boot Acknowledge Disable / Pull-Down During Power Cycle Enable	Yes	0	MMC 0 - Boot Acknowledge Enabled. 1 - Boot Acknowledge Disabled. SD 0 - Use default SD pad settings during power cycle 1 - Set pull-down on SD pads during power cycle (used only if "SD Power Cycle Enable" enabled)
BOOT_CFG2[0]	OEM	Override Pad Settings	Yes	0	0 - Use default values 1 - Use PAD_SETTINGS values
MMC_DLL_DLY[6:0]	OEM	MMC DLL Value / UHSI Calibration Start Value	No	0000000	MMC DLL Value / UHSI Calibration Start Value

1. Setting can be overridden by GPIO settings when BT\_FUSE\_SEL fuse is intact. See [Table 1](#) for corresponding GPIO pin.

Boot code supports following standards.

- MMCv4.4 or less
- eMMCv4.4 or less
- SDv2.0 or less
- eSDv2.10 rev-0.9, with or without FAST\_BOOT.
- SDXCv3.0

MMC/SD/eSD/SDXC/eMMC can be connected to any of USDHC-1,2,3,4 blocks and can be booted by copying 4Kbyte of data from MMC/SD/eSD/eMMC device to internal RAM. After checking the Image Vector Table header value (0xD1) from Program Image, the ROM code performs a DCD check. After successful DCD extraction, the ROM code extracts from Boot Data Structure the destination pointer and length of image to be copied to RAM device from where code execution occurs.

The maximum image size to load in SD/MMC boot is 32MB. This is due to the limited number of uSDHC ADMA Buffer Descriptors allocated by ROM.

**NOTE**

The Initial 4Kbyte of Program Image must contain the IVT, DCD and the Boot Data structures.

**Table 8-12. SD/MMC Frequencies**

	SD	MMC	MMC (DDR Mode)
Identification (KHz)	347.22		
Normal Speed Mode (MHz)	25	20	25
High Speed Mode (MHz)	50	40	50
UHSI SDR50 (MHz)	100		
UHSI SDR104 (MHz)	200		

**NOTE**

The boot ROM code reads application image length and application destination pointer from image.

**8.5.2.2 MMC and eMMC Boot**

The following table provides MMC and eMMC boot details.

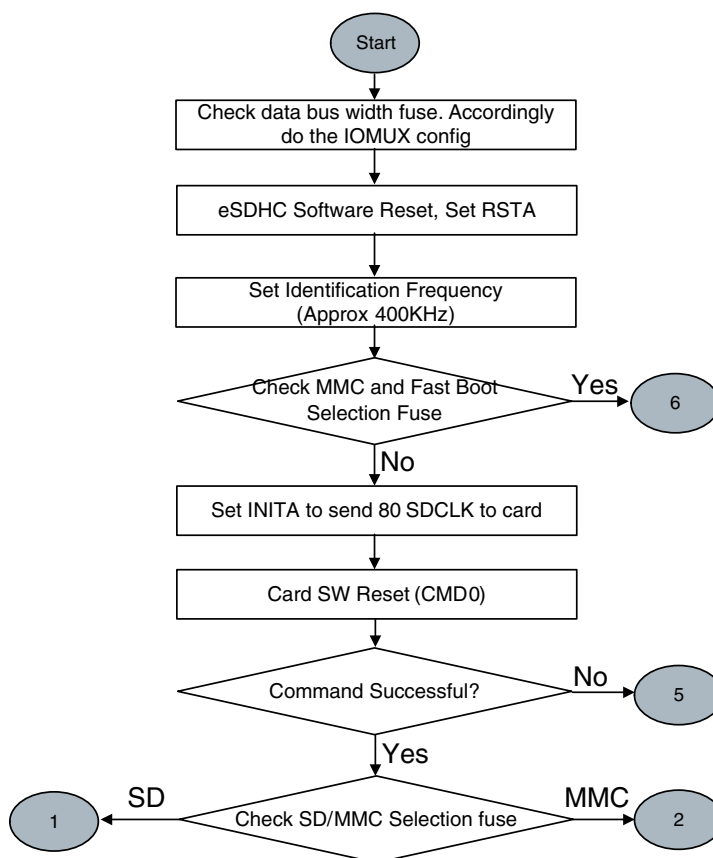
**Table 8-13. MMC and eMMC Boot Details**

Normal Boot Mode	<p>During initialization (normal boot mode) the MMC frequency is set to 347.22 KHz. When the MMC card enters the identification portion of the initialization, voltage validation is performed and the ROM boot code checks high voltage settings and card capacity. The ROM boot code supports both high capacity and low capacity MMC/eMMC cards. After initialization phase is complete, the ROM boot code switches to a higher frequency (20 MHz in Normal boot mode or 40MHz in High Speed mode). eMMC is also interfaced via USDHC and follows the same flow as MMC.</p> <p>The boot partition can be selected for an MMC4.x card after the card initialization is complete. The ROM code reads the BOOT_PARTITION_ENABLE field in the Ext_CSD[179] to get the boot partition to be set. If there is no boot partition mentioned in BOOT_PARTITION_ENABLE field or the user partition has been mentioned, ROM boots from the user partition.</p>
------------------	--

*Table continues on the next page...*

**Table 8-13. MMC and eMMC Boot Details (continued)**

eMMC4.3 or eMMC4.4 Device Supporting Special Boot Mode	If using an eMMC4.3 or eMMC4.4 device supporting special boot mode, it can be initiated by pulling the CMD line low. If BOOT ACK is enabled, the eMMC4.3/eMMC4.4 device sends the BOOT ACK via DATA lines and ROM can read the BOOT ACK [S010E] to identify the eMMC4.3/eMMC4.4 device. eMMC4.3/eMMC4.4 device with "Boot mode" feature can only be supported via ESDHCV3-3 and with or without BOOT ACK. If BOOT ACK is enabled ROM waits 50 ms to get the BOOT ACK and if BOOT ACK is received by ROM. If BOOT ACK is disabled ROM waits 1 second for data. If BOOT ACK or data was received then eMMC4.3/eMMC4.4 is booted in "Boot mode", otherwise eMMC4.3/eMMC4.4 boots as a normal MMC card from the selected boot partition. This boot mode can be selected by BOOT_CFG1[4] (Fast Boot) fuse. BOOT ACK is selected by BOOT_CFG2[1].
eMMC4.4 Device	If using eMMC4.4 device, Double Data Rate (DDR) mode can be used. This mode can be selected by BOOT_CFG2[7:5] (Bus Width) fuse.

**Figure 8-5. Expansion Device Boot Flow (1 of 6)**

## Boot Devices (Internal Boot)

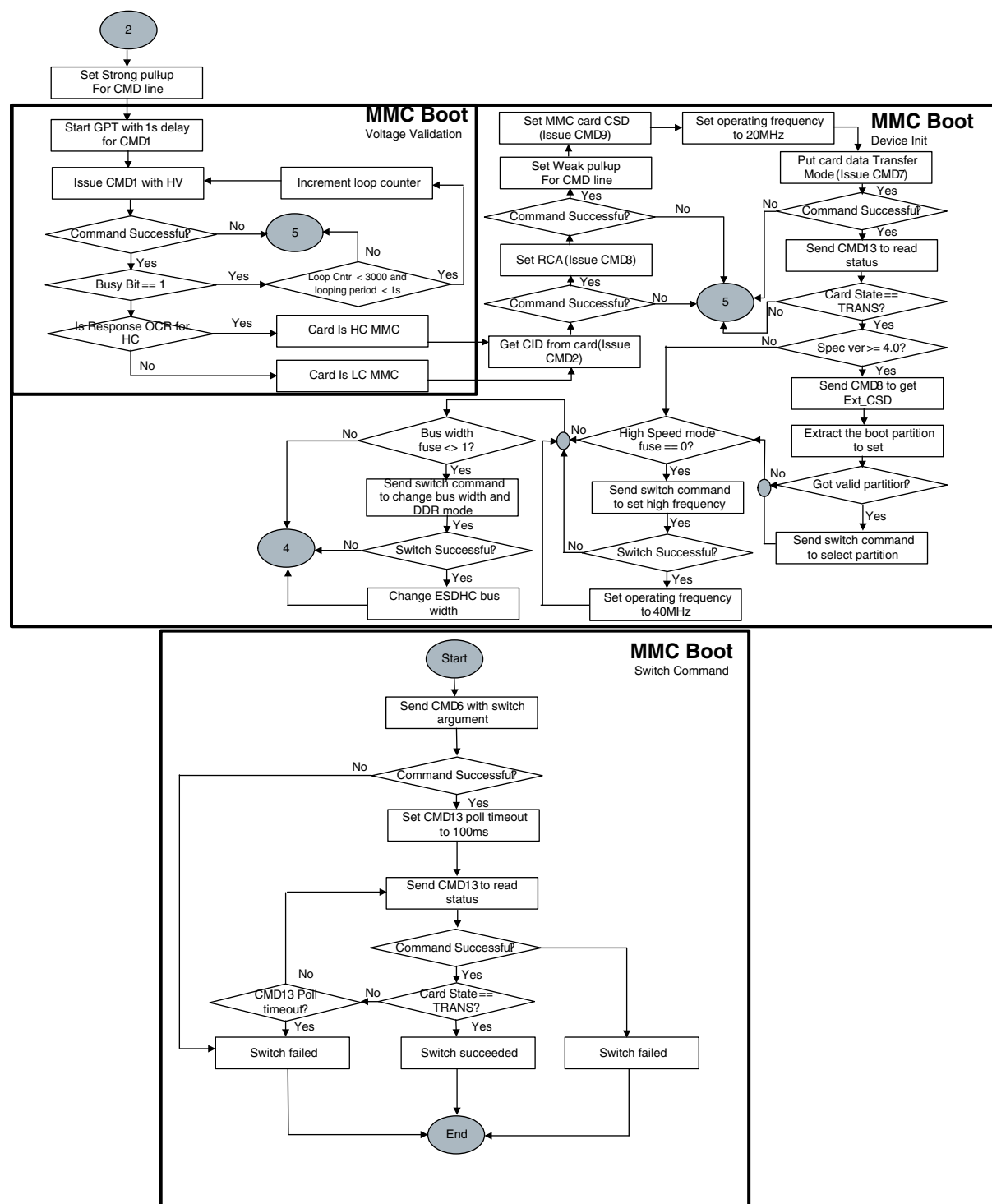


Figure 8-6. Expansion Device (MMC) Boot Flow (2 of 6)

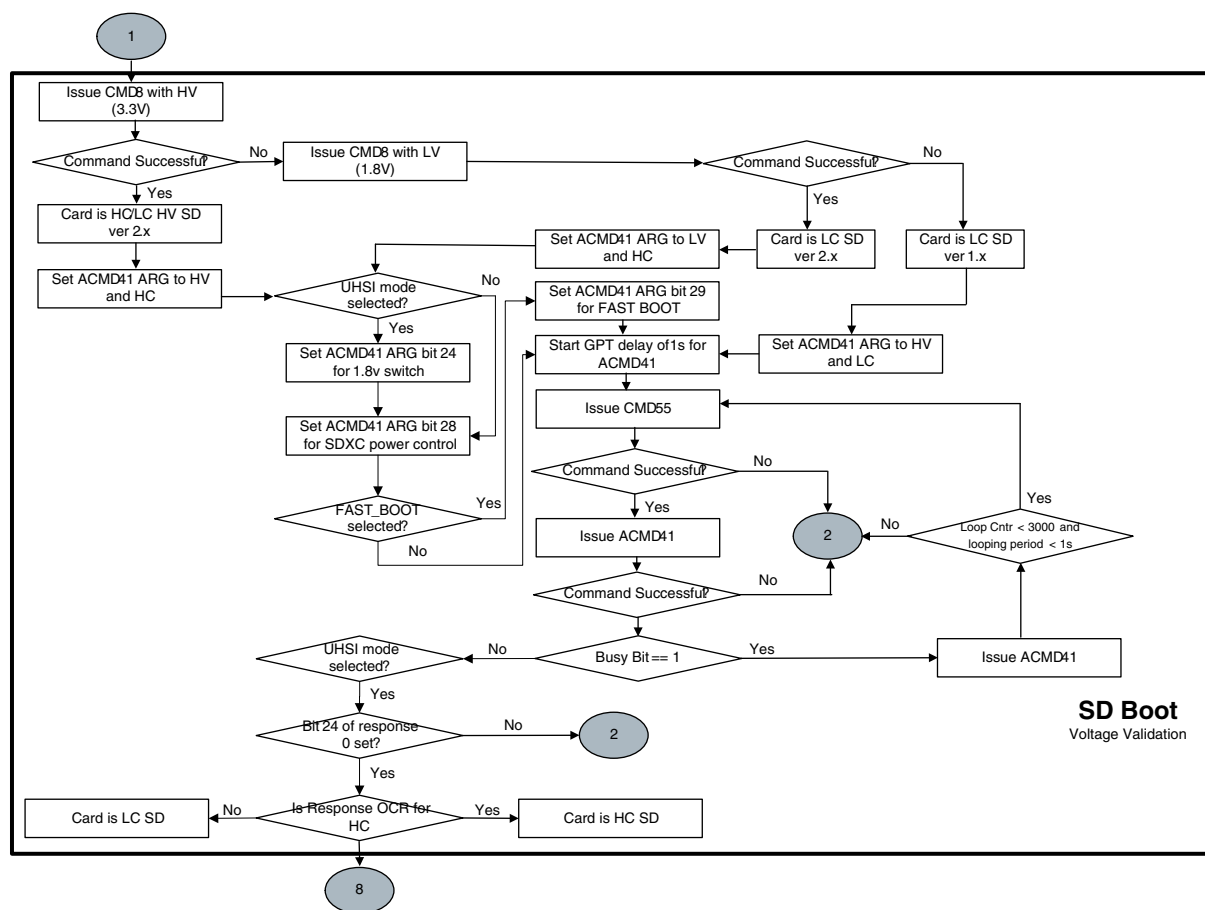


Figure 8-7. Expansion Device (SD/eSD/SDXC) Boot Flow (3 of 6) Part 1

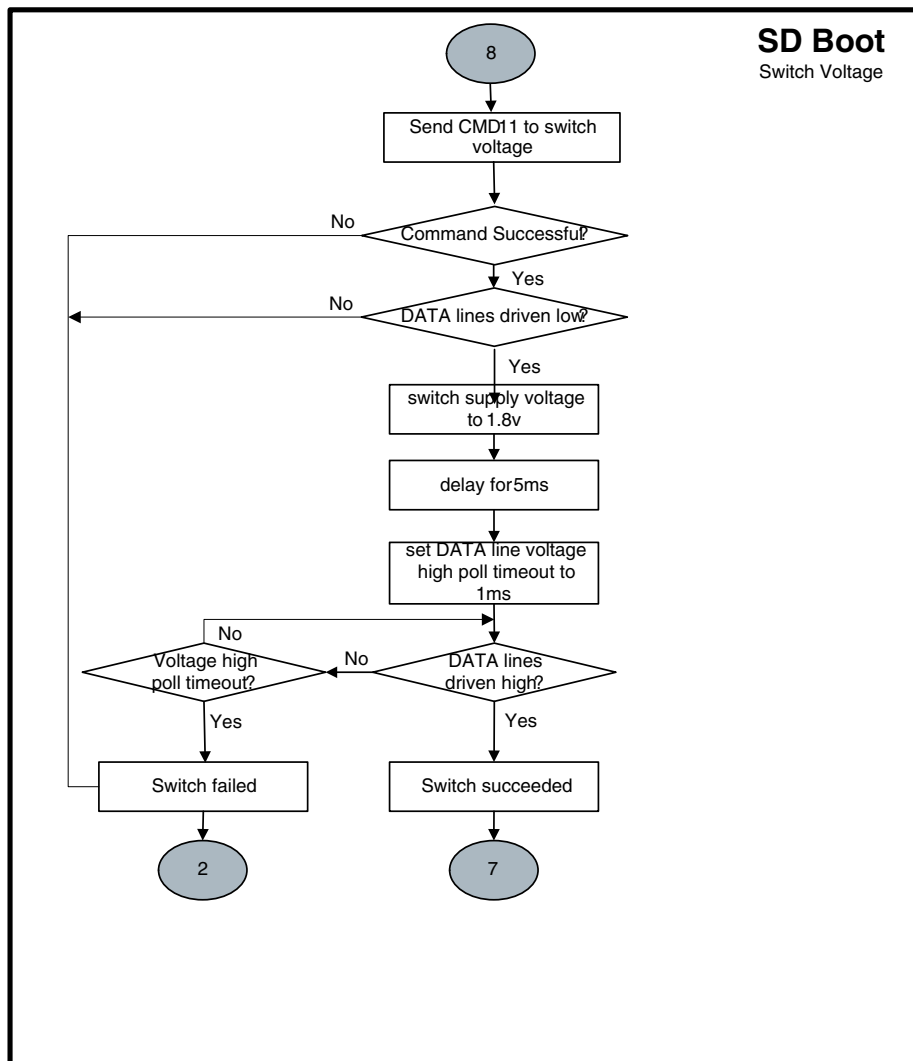
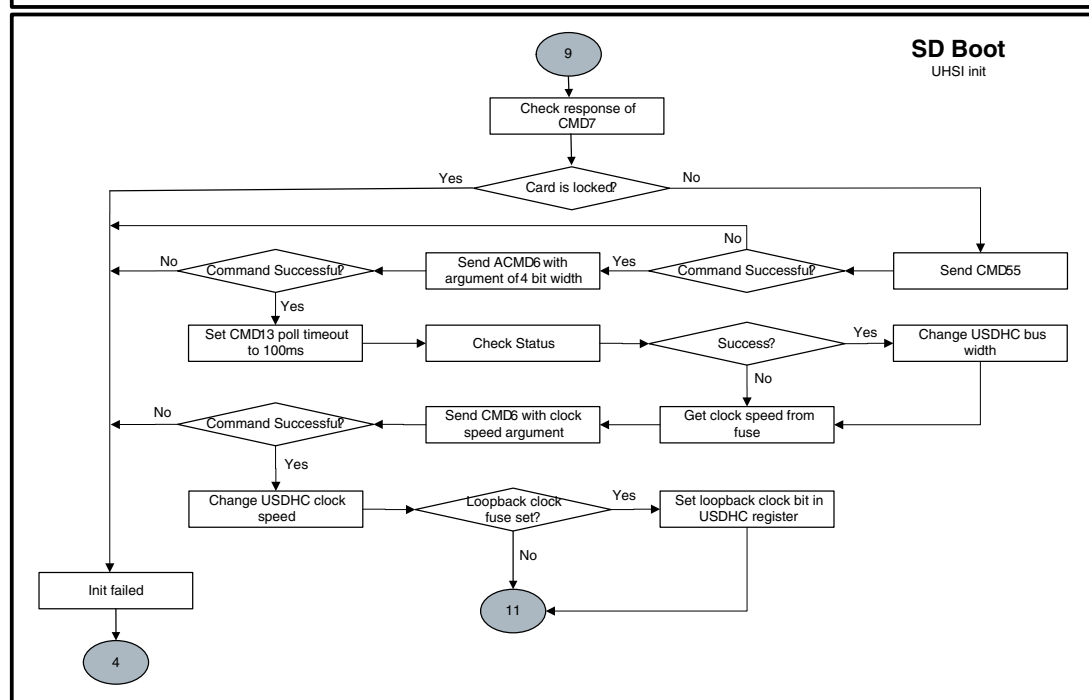
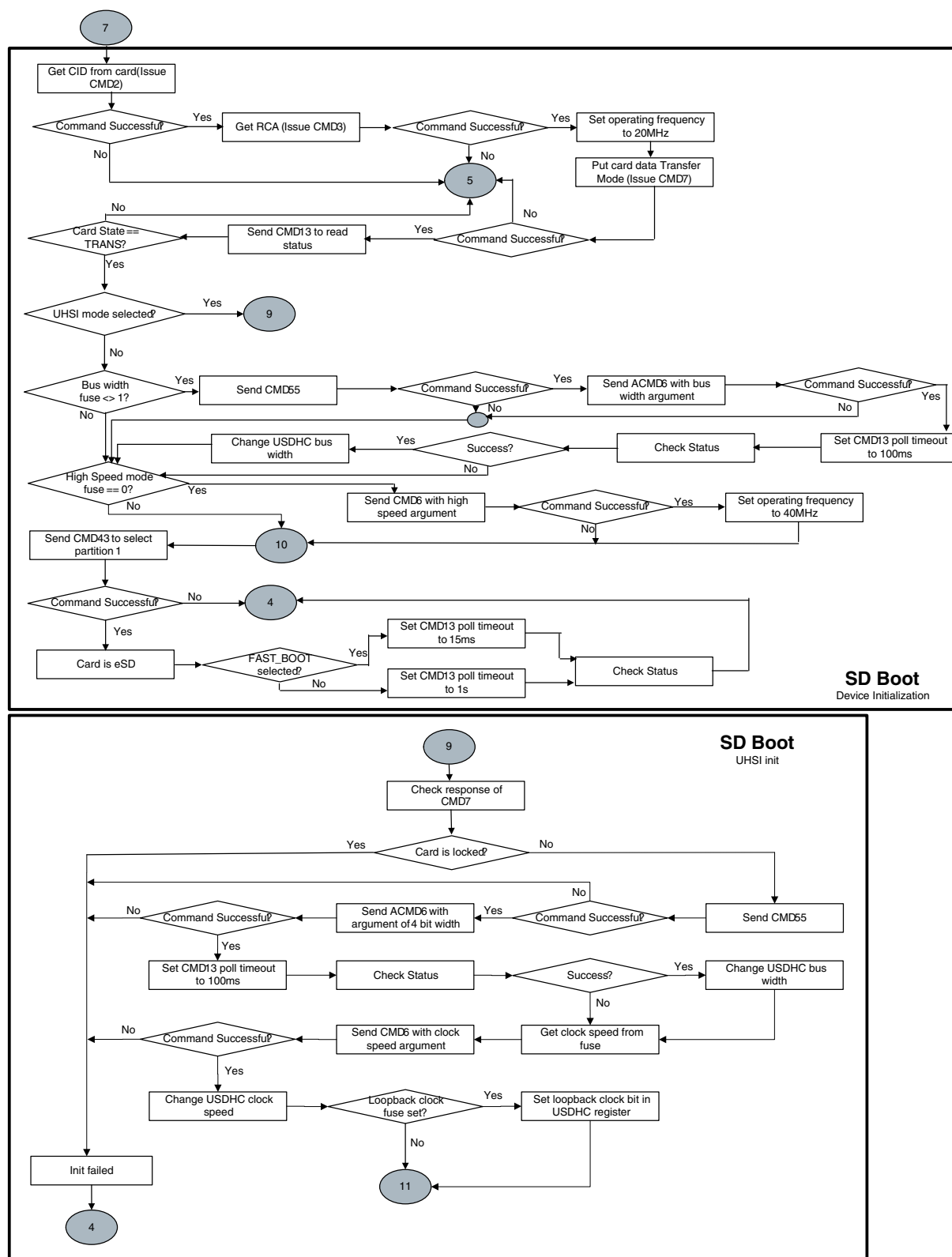


Figure 8-8. Expansion Device (SD/eSD/SDXC) Boot Flow (3 of 6) Part 2





**Figure 8-9. Expansion Device (MMCSD/eSD/SDXC) Boot Flow (4 of 6)**  
i.MX 6SoloLite Applications Processor Reference Manual, Rev. 1, 04/2013

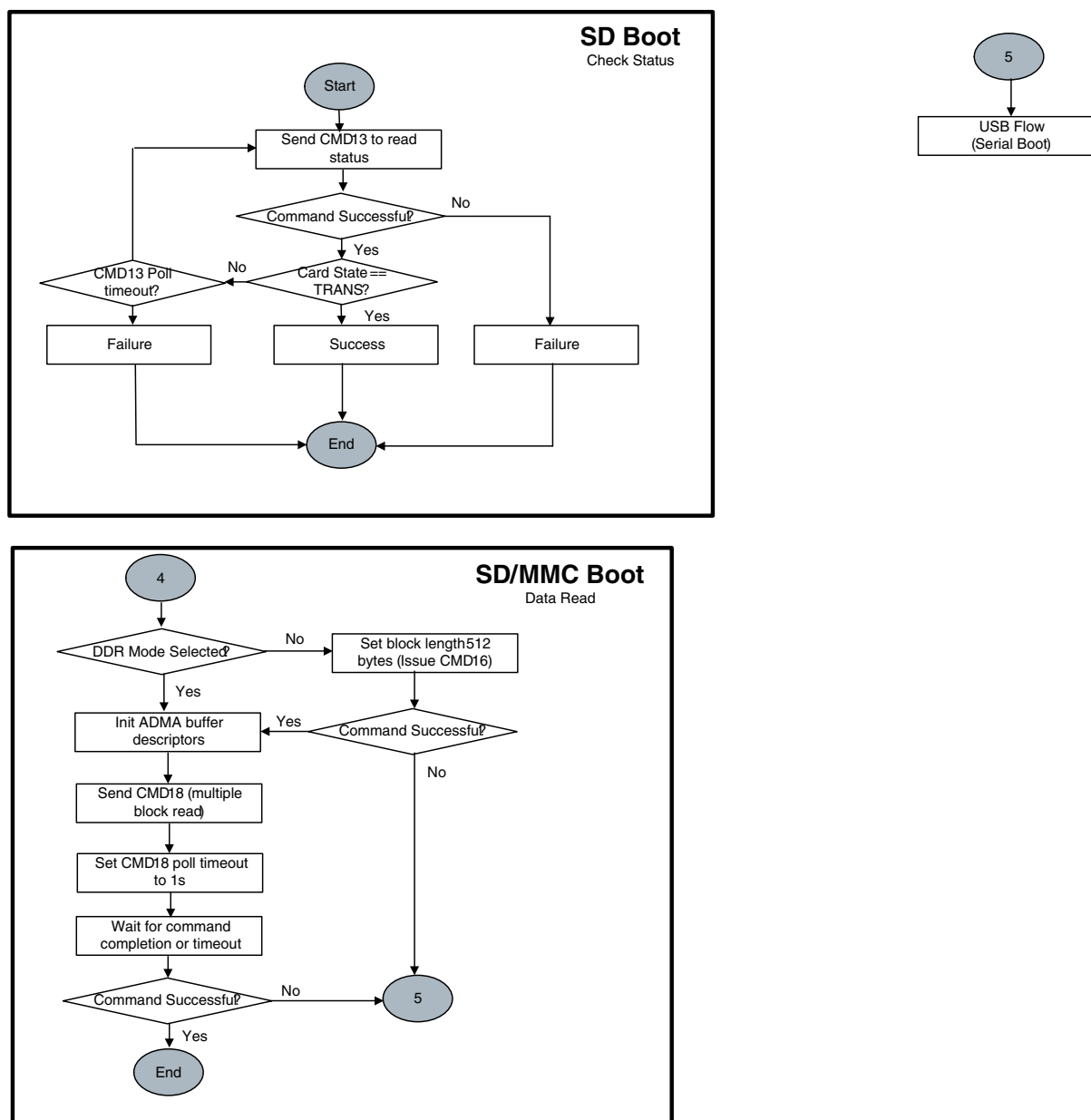
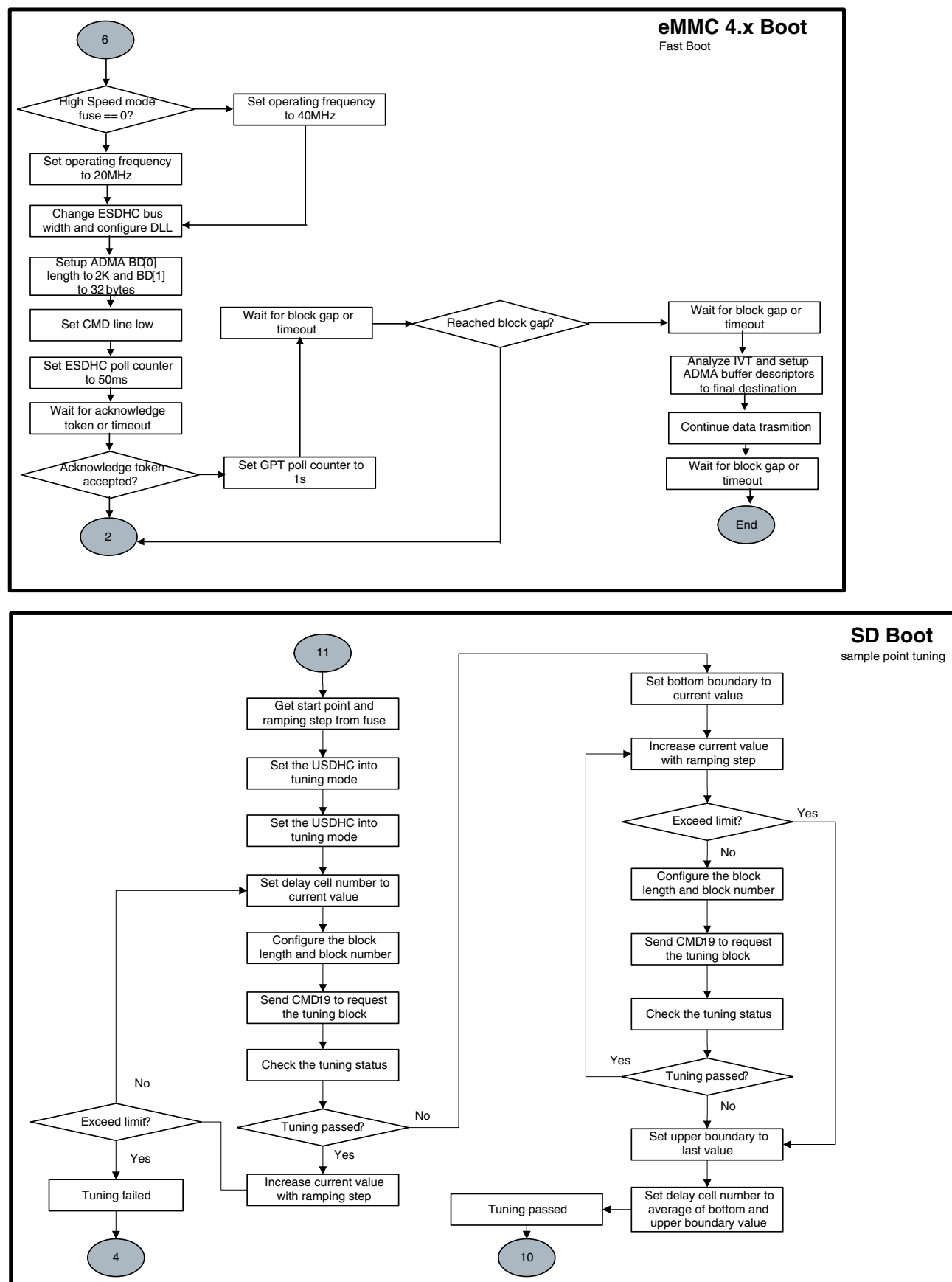


Figure 8-10. Expansion Device (SD/eSD) Boot Flow (5 of 6)



**Figure 8-11. Expansion Device Boot Flow (6 of 6)**  
i.MX 6SoloLite Applications Processor Reference Manual, Rev. 1, 04/2013

### 8.5.2.3 SD, eSD and SDXC

After the normal boot mode initialization begins, the SD/eSD/SDXC frequency is set to 347.22 kHz. During the identification phase, SD/eSD/SDXC card voltage validation is performed. During voltage validation, boot code first checks with high voltage settings; if that fails, it checks with low voltage settings.

The capacity of the card is also checked. Boot code supports high capacity and low capacity SD/eSD/SDXC cards after voltage validation card initialization is done.

During card initialization, the ROM boot code attempts to set the boot partition for all SD, eSD, and SDXC devices. If this fails, the boot code assumes the card is a normal SD card or SDXC card. If it does not fail, the boot code assumes it is an eSD card. After the initialization phase is over, boot code switches to a higher frequency (25 MHz in Normal Speed mode or 50 MHz in High Speed Mode). ROM also supports FAST\_BOOT mode booting from eSD card. This mode can be selected by BOOT\_CFG1[4] (Fast Boot) fuse described in [Table 8-11](#).

For UHSI cards, clock speed fuses can be set to SDR50 or SDR104 on USDHC3 and USDHC4 ports. This will enable the voltage switch process to set the signaling voltage to 1.8V during voltage validation. The bus width is fixed at 4 bits wide and a sampling point tuning process is needed to calibrate the number of delay cells. If SD Loopback Clock eFuse is set, the feedback clock will come directly from the loopback SD clock, instead of the card clock (by default). The SD clock speed can be selected by BOOT\_CFG1[3:2], and the SD Loopback Clock is selected by BOOT\_CFG1[0].

UHSI calibration start value (MMC\_DLL\_DLY[6:0]) and step value (BOOT\_CFG2[7:5]) can be set to optimize the sample point tuning process.

If SD Power Cycle Enable eFuse is 1, ROM will set SD\_RST pad low, wait 5ms and then set SD\_RST pad high. If SD\_RST pad is connected to SD power supply enable logic on board, it enables power cycle of SD card. This may be crucial in case when SD logic is in 1.8V states and must be reset to 3.3V states.

### 8.5.2.4 IOMUX Configuration for SD/MMC

**Table 8-14. SD/MMC IOMUX Pin Configuration**

Signal	USDHC-1	USDHC-2	USDHC-3	USDHC-4
<b>CLK</b>	SD1_CLK.alt0	SD2_CLK.alt0	SD3_CLK.alt0	FEC_MDIO.alt1
<b>CMD</b>	SD1_CMD.alt0	SD2_CMD.alt0	SD3_CMD.alt0	FEC_TX_CLK.alt1
<b>DAT0</b>	SD1_DAT0.alt0	SD2_DAT0.alt0	SD3_DAT0.alt0	FEC_RX_ER.alt1

*Table continues on the next page...*

**Table 8-14. SD/MMC IOMUX Pin Configuration (continued)**

Signal	USDHC-1	USDHC-2	USDHC-3	USDHC-4
<b>DAT1</b>	SD1_DAT1.alt0	SD2_DAT1.alt0	SD3_DAT1.alt0	FEC_CRSDV.alt1
<b>DAT2</b>	SD1_DAT2.alt0	SD2_DAT2.alt0	SD3_DAT2.alt0	FEC_RXD1.alt1
<b>DAT3</b>	SD1_DAT3.alt0	SD2_DAT3.alt0	SD3_DAT3.alt0	FEC_TXD0.alt1
<b>DAT4</b>	SD1_DAT4.alt0	SD2_DAT4.alt0	KEY_COL1.alt4	FEC_MDC.alt1
<b>DAT5</b>	SD1_DAT5.alt0	SD2_DAT5.alt0	KEY_ROW1.alt4	FEC_RXD0.alt1
<b>DAT6</b>	SD1_DAT6.alt0	SD2_DAT6.alt0	KEY_COL2.alt4	FEC_TX_EN.alt1
<b>DAT7</b>	SD1_DAT7.alt0	SD2_DAT7.alt0	KEY_ROW2.alt4	FEC_TXD1.alt1
<b>VSELECT</b>	ECSP12_MOSI.alt4	ECSP11_MOSI.alt4	KEY_ROW6.alt6	EPDC_PWCTRL1.alt6
<b>RESET</b> <sup>1</sup>	KEY_COL3.alt5	SD2_RESET.alt0	KEY_COL6.alt5	FEC_REFOUT.alt1
<b>CD</b>	KEY_ROW7.alt6	-	-	EPDC_PWRCTRL3.alt6

1. Active low

### 8.5.2.5 Redundant Boot Support for Expansion Device

ROM supports redundant boot for expansion device. Primary or Secondary image is selected depending on PERSIST\_SECONDARY\_BOOT setting (see [Table 8-7](#)).

If PERSIST\_SECONDARY\_BOOT is 0, the boot ROM uses address 0x0 for primary image.

If PERSIST\_SECONDARY\_BOOT is 1, the boot ROM will read secondary image table from address 0x200 on boot media and will use address specified in the table.

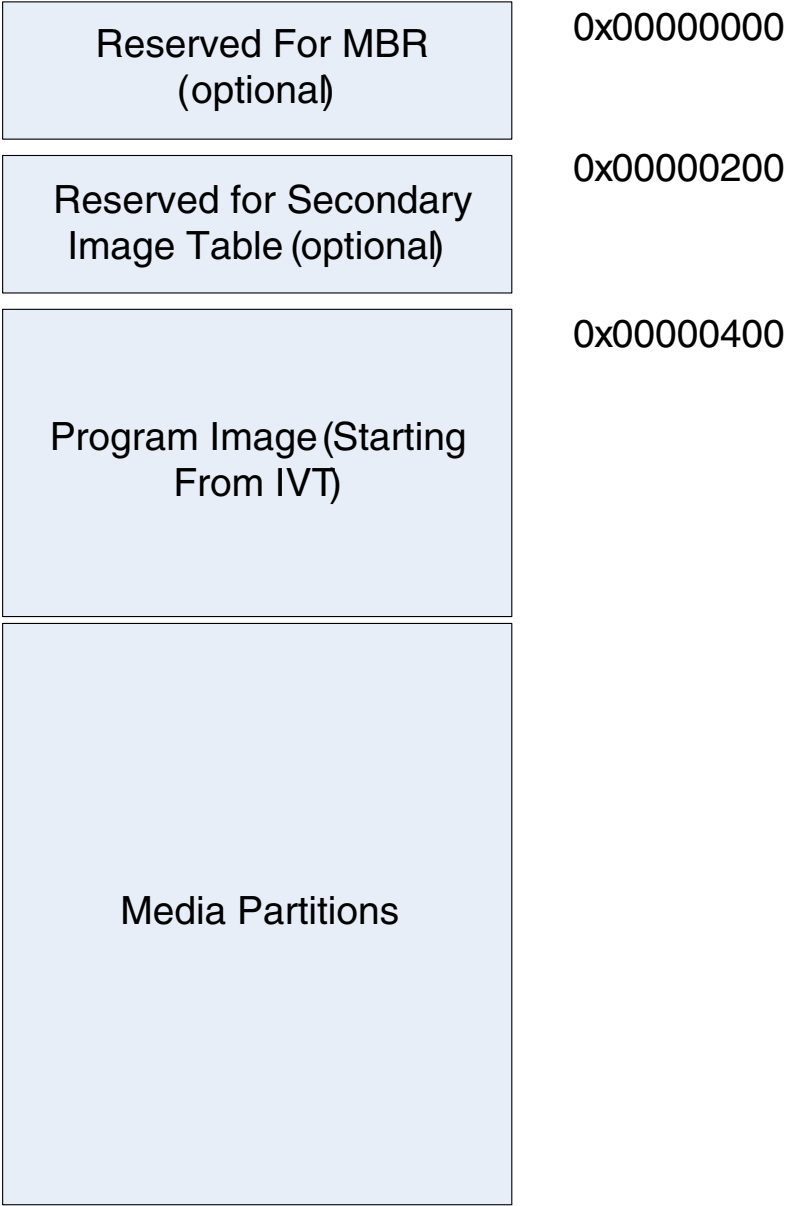
**Table 8-15. Secondary Image Table Format**

Reserved (chipNum)
Reserved (driveType)
tag
firstSectorNumber
Reserved (sectorCount)

Where:

- tag: used as indication of valid secondary image table. Must be 0x00112233.
- firstSectorNumber is the first 512B sector number of the secondary image.

For secondary image support, the primary image must reserve space for secondary image table. See the figure below for typical structures layout on expansion device.



**Figure 8-12. Expansion Device Structures Layout**

For Closed mode, if there are failures during primary image authentication, the boot ROM will turn on PERSIST\_SECONDARY\_BOOT bit (see [Table 8-7](#)) and perform software reset. (After software reset, secondary image will be used.)

### 8.5.3 Serial ROM through SPI and I2C

The chip supports boot from serial memory devices, such as EEPROM and Serial Flash using the SPI.

The following ports are available for serial boot: ECSPI ( ECSPI-1, ECSPI-2, ECSPI-3, ECSPI-4, ECSPI-5), and I2C Controller ( I2C-1, I2C-2 and I2C-3) interfaces.

#### 8.5.3.1 Serial ROM eFUSE Configuration

The boot ROM code determines the type of device using the following parameters, either provided by eFUSE settings or sampled on the I/O pins, during boot.

See the table below for details:

**Table 8-16. Serial ROM Boot eFUSE Descriptions**

Fuse	Config	Definition	GPIO <sup>1</sup>	Shipped Value	Settings
BOOT_CFG1[7:4]	OEM	Boot Device Selection	Yes	0000	0011 - Boot from Serial ROM
BOOT_CFG4[5:4]	OEM	CS select (SPI only)	Yes	00	00 - ECSPiX_SS0 01 - ECSPiX_SS1 10 - ECSPiX_SS2 11 - ECSPiX_SS3
BOOT_CFG4[3]	OEM	SPI Addressing (SPI only)	Yes	0	0 - 2-bytes (16-bit) 1 - 3-bytes (24-bit)
BOOT_CFG4[2:0]	OEM	Port Select	Yes	00	000 - ECSPI-1 001 - ECSPI-2 010 - ECSPI-3 011 - ECSPI-4 101- I2C-1 110- I2C-2 111- I2C-3

1. Setting can be overridden by GPIO settings when BT\_FUSE\_SEL fuse is intact. See [Table 1](#) for corresponding GPIO pin.

The ECPSI-1/ECPSI-2/ECPSI-3/ECPSI-4 block can be used as boot device using ECSPI interface for serial ROM boot. The SPI interface is configured to operate at 15MHz for 3-byte addressing device and 3.75MHz for 2-byte addressing devices.

The I2C-1/I2C-2/I2C-3 block can be used as boot device using I2C interface, for serial ROM boot. The I2C interface is configured to operate at 343.75 Kbps.

The boot ROM will copy 4Kbyte of data from Serial ROM device to internal RAM. After checking the Image Vector Table header value (0xD1) from Program Image, the ROM code performs a DCD check. After successful DCD extraction, the ROM code extracts from Boot Data Structure the destination pointer and length of image to be copied to RAM device from where code execution occurs.

### NOTE

The Initial 4K of Program Image must contain the IVT, DCD and the Boot Data structures.

### 8.5.3.2 I2C Boot

The boot flow when booting from an I2C device is shown in [Figure 8-13](#).

The boot ROM code reads the fuses BOOT\_CFG1[7:4] (Boot Device Selection) and BOOT\_CFG1[7:4] (Port select) to detect EEPROM device type. The ROM program copies 4K data from the EEPROM device to internal RAM. The boot ROM code next copies the initial 4Kbyte of data as well as rest of image directly to application destination extracted from application image.

The chip uses the Device Select Code/Device Address in the table below to boot from an EEPROM.

**Table 8-17. EEPROM via I2C Device Select Code**

Bits	Device Type Identifier				Chip Enable Address <sup>1</sup>			R/W
	7	6	5	4	3	2	1	0
Device Select Code	1	0	1	0	0	0	0	R/W

1. These address bits, should be configured at the memory device, to match this '000' value.



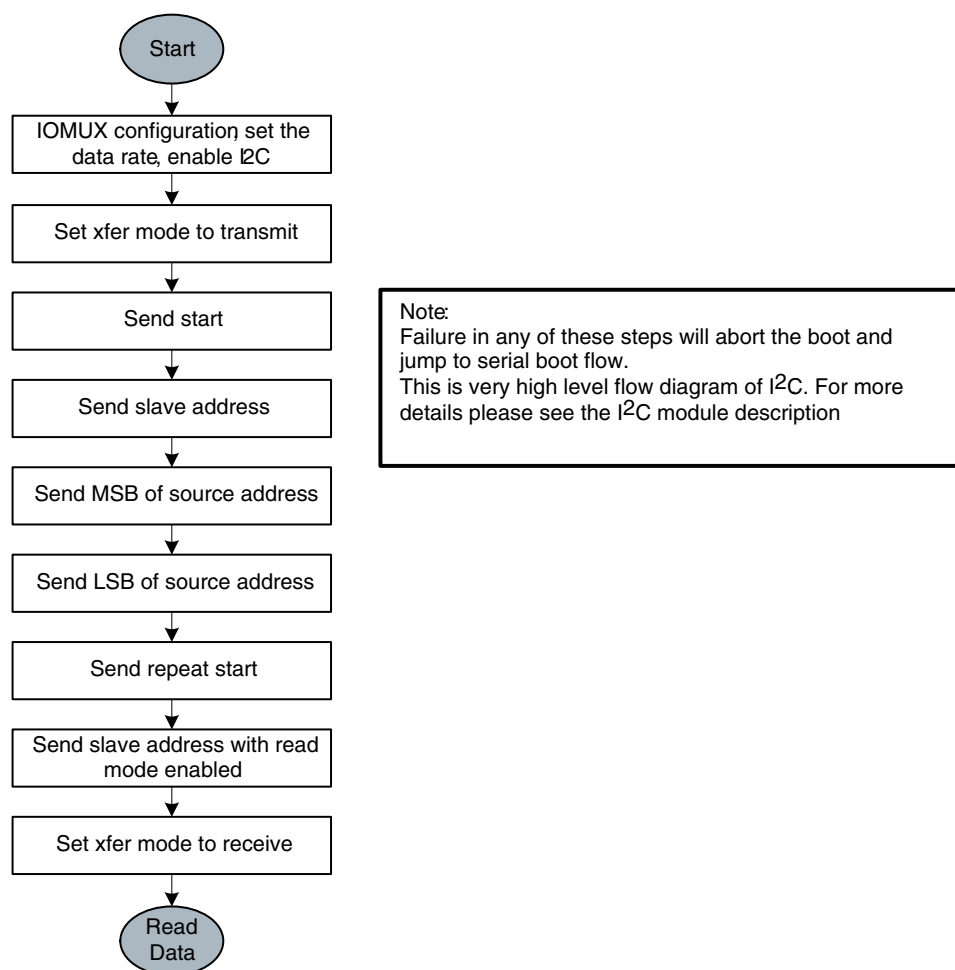


Figure 8-13. I2C Flow Chart

### 8.5.3.2.1 I2C IOMUX Pin Configuration

The contacts assigned to the signals used by the three I2C blocks is shown in the table below.

Table 8-18. I2C IOMUX Pin Configuration

Signal	I2C-1	I2C-2	I2c-3
SDA	I2C1_SCL.alt0	I2C2_SCL.alt0	AUD_RXFS.alt4
SCL	I2C1_SDA.alt0	I2C2_SDA.alt0	AUD_RXC.alt4

### 8.5.3.3 ECSPI Boot

The Enhanced Configurable SPI (ECSPI) interface is configured in master mode and the EEPROM device is connected to ECSPI interface as a slave.

The boot ROM code copies 4 KB data from EEPROM device to the internal RAM. If DCD verification is successful, the ROM code copies the initial 4 KB data, as well as the rest of the image extracted from application image, directly to the application destination. The ECSPI can read data from EEPROM using 2 or 3 byte addressing. Its burst length is 32 bytes.

#### NOTE

The Serial ROM Chip Select Number is determined by  
BOOT\_CFG4[5:4] (Chip Select) fuse.

When using the SPI as boot device, the Chip supports booting from both Serial EEPROM and Serial Flash devices. The boot code determines which device is being used by reading the appropriate eFUSE/I/O values at boot (see [Table 8-16](#) for details).

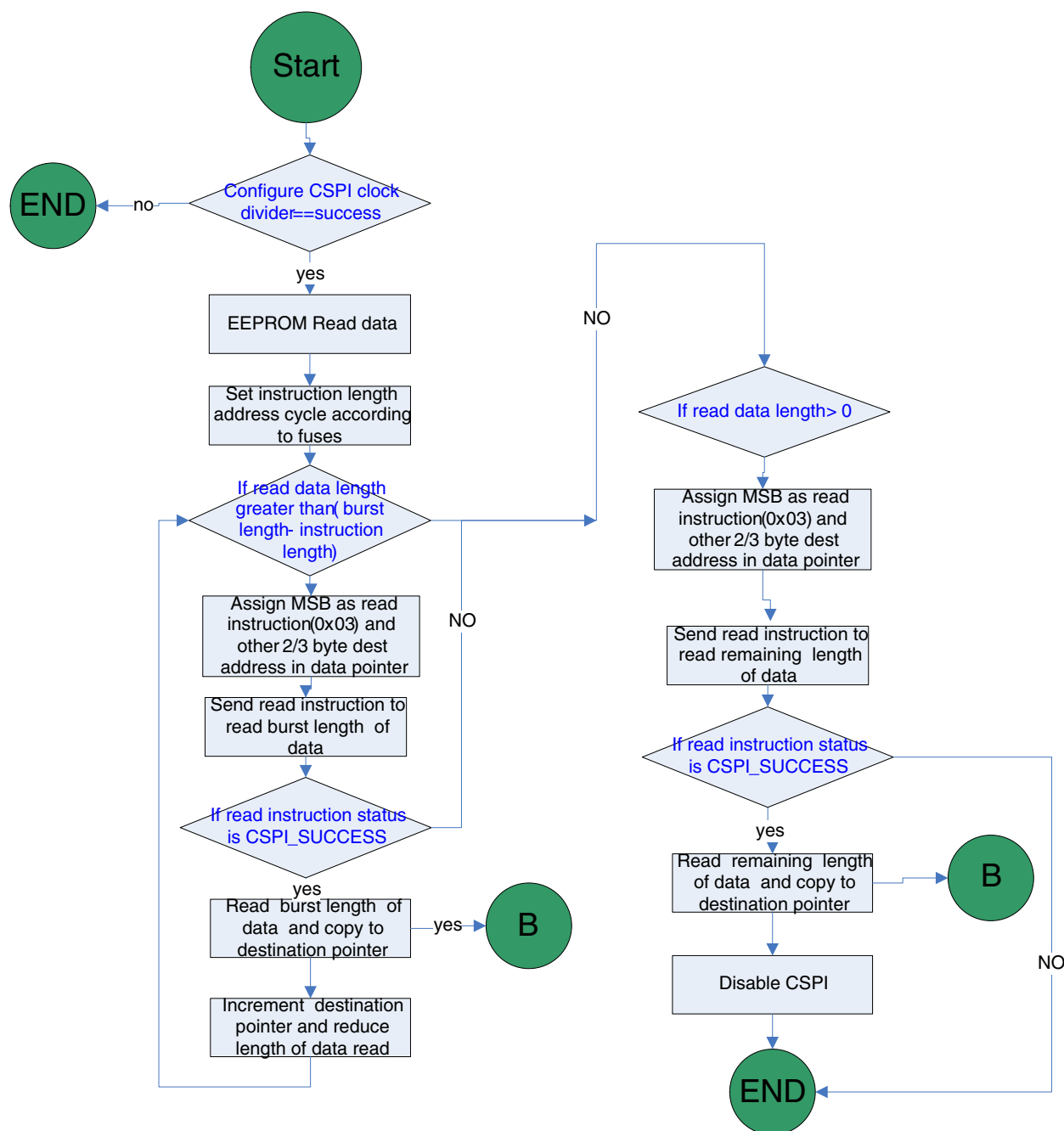


Figure 8-14. CSPI Flow chart

### 8.5.3.3.1 ECSPI IOMUX Pin Configuration

The contacts assigned to the signals used by the three CSPI blocks is shown in the table below.

**Table 8-19. SPI IOMUX Pin Configuration**

Signal	ECSPI-1	ECSPI-2	ECSPI-3	ECSPI4	ECSPI-5
<b>MISO</b>	ECSPI1_MISO.alt0	ECSPI2_MISO.alt0	EPDC_D9.alt1	EPDC_D1.alt1	N/A
<b>MOSI</b>	ECSPI1_MOSI.alt0	ECSPI2_MOSI.alt0	EPDC_D8.alt1	EPDC_D0.alt1	N/A
<b>RDY</b>	N/A <sup>1</sup>	N/A	N/A	N/A	N/A
<b>SCLK</b>	ECSPI1_SCLK.alt0	ECSPI2_SCLK.alt0	EPDC_D11.alt1	EPDC_D3.alt1	N/A
<b>SS0</b>	ECSPI1_SS0.alt0	ECSPI2_SS0.alt0	EPDC_D10.alt1	EPDC_D2.alt1	N/A
<b>SS1</b>	I2C1_SCL.alt6	EPDC_SDCE0.alt1	EPDC_D12.alt6	EPDC_D4.alt1	N/A
<b>SS2</b>	I2C1_SDA.alt6	EPDC_GDCLK.alt1	EPDC_D13.alt6	EPDC_D5.alt1	N/A
<b>SS3</b>	ECSPI2_SS0.alt1	EPDC_GDOE.alt1	EPDC_D14.alt6	EPDC_D6.alt1	N/A

1. N/A in the ROM code indicates the pins are not available or not used.

## 8.6 Program image

This section describes the data structures that are required to be included in a user's program image. A program image consists of:

- Image vector table—A list of pointers located at a fixed address that the ROM examines to determine where other components of the program image are located
- Boot data—A table indicating the program image location, program image size in bytes, and the plugin flag
- Device configuration data—IC configuration data
- User code and data

### 8.6.1 Image Vector Table and Boot Data

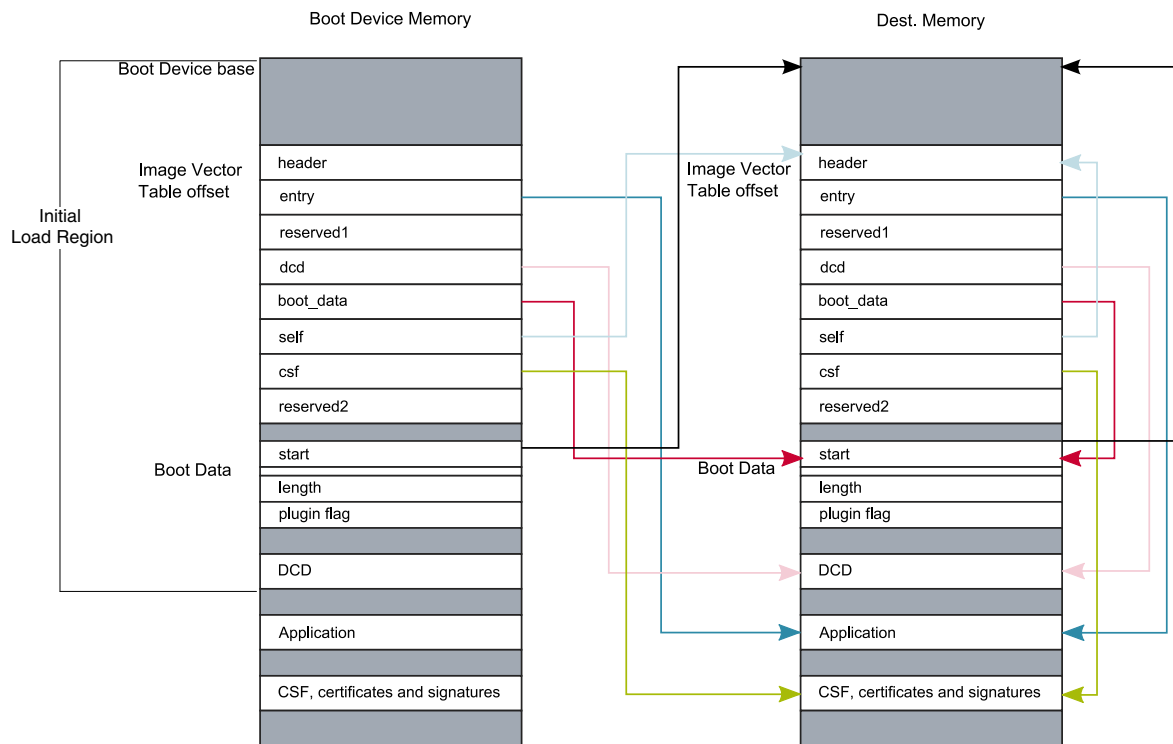
The Image Vector Table (IVT) is the data structure that the ROM reads from the boot device supplying the program image containing the required data components to perform a successful boot.

The IVT includes the program image entry point, a pointer to Device Configuration Data (DCD) and other pointers used by the ROM during the boot process. The ROM locates the IVT at a fixed address that is determined by the boot device connected to the Chip. The IVT offset from the base address and initial load region size for each boot device

type is defined in the table below. The location of the IVT is the only fixed requirement by the ROM. The remainder of the image memory map is flexible and is determined by the contents of the IVT.

**Table 8-20. Image Vector Table Offset and Initial Load Region Size**

Boot Device Type	Image Vector Table Offset	Initial Load Region Size
NOR	4 Kbyte = 0x1000 bytes	Entire Image Size
OneNAND	256 bytes = 0x100 bytes	1 Kbyte
SD/MMC/eSD/eMMC/SDXC	1 Kbyte = 0x400 bytes	4 Kbyte
I2C/SPI EEPROM	1 Kbyte = 0x400 bytes	4 Kbyte



**Figure 8-15. Image Vector Table**

### 8.6.1.1 Image Vector Table Structure

The IVT has the following format where each entry is a 32 bit word:

**Table 8-21. IVT Format**

header
--------

*Table continues on the next page...*

**Table 8-21. IVT Format (continued)**

entry:	Absolute address of the first instruction to execute from the image
reserved1:	Reserved and should be zero
dcd:	Absolute address of the image DCD. The DCD is optional so this field may be set to NULL if no DCD is required. See <a href="#">Device Configuration Data (DCD)</a> for further details on DCD.
boot data:	Absolute address of the Boot Data
self:	Absolute address of the IVT. Used internally by the ROM
csf:	Absolute address of Command Sequence File (CSF) used by the HAB library. See <a href="#">High Assurance Boot (HAB)</a> for details on secure boot using HAB. This field must be set to NULL when not performing a secure boot
reserved2:	Reserved and should be zero

The IVT header has the following format:

**Table 8-22. IVT Header Format**

Tag	Length	version
-----	--------	---------

where:

Tag: A single byte field set to 0xD1

Length: a two byte field in big endian format containing the overall length of the IVT, in bytes, including the header. (the length is fixed and must have a value of 32 bytes)

Version: A single byte field set to 0x40 or 0x41

### 8.6.1.2 Boot Data Structure

The Boot Data must follow the format defined in the table found here, each entry is a 32-bit word.

**Table 8-23. Boot Data Format**

start	Absolute address of the image
length	Size of the program image
plugin	Plugin flag (see <a href="#">Plugin Image</a> )

## 8.6.2 Device Configuration Data (DCD)

Upon reset, the Chip uses the default register values for all peripherals in the system. However, these settings typically are not ideal for achieving optimal system performance and there are even some peripherals that must be configured before they can be used.

The DCD is configuration information contained in a Program Image, external to the ROM, that the ROM interprets to configure various peripherals on the Chip.

For example, the EIM default settings allow the core to interface to a NOR flash device immediately out of reset. This allows the Chip to interface with any NOR flash device, but has the cost of slow performance. Additionally, some components such as DDR require some sequence of register programming as part of configuration before it is ready to be used. The DCD feature can be used to program the EIM registers and MMDC registers to the optimal settings.

The ROM determines the location of the DCD table based on information located in the Image Vector Table (IVT). See [Image Vector Table and Boot Data](#) for more details. The DCD table shown below is a big endian byte array of the allowable DCD commands. The maximum size of the DCD limited to 1768 bytes.

**Table 8-24. DCD Data format**

Header
[CMD]
[CMD]
...

The DCD header is 4 bytes with the following format:

**Table 8-25. DCD Header**

Tag	Length	Version
-----	--------	---------

where:

Tag: A single byte field set to 0xD2

Length: a two byte field in big endian format containing the overall length of the DCD, in bytes, including the header

Version: A single byte field set to 0x41

### 8.6.2.1 Write Data Command

The Write Data Command is used to write a list of given 1-, 2- or 4-byte values or bitmasks to a corresponding list of target addresses.

The format of Write Data Command, again a big endian byte array, is shown in the table below.

**Table 8-26. Write Data Command Format**

Tag	Length	Parameter
-----	--------	-----------

*Table continues on the next page...*

**Table 8-26. Write Data Command Format (continued)**

Address
Value/Mask
[Address]
[Value/Mask]
...
[Address]
[Value/Mask]

where:

Tag: A single byte field set to 0xCC

Length: A two byte field in big endian format containing the length of the Write Data Command, in bytes, including the header

Address: target address to which data should be written

Value/Mask: data value or bitmask to be written to preceding address

The Parameter field is a single byte divided into bitfields as follows:

**Table 8-27. Write Data Command Parameter field**

7	6	5	4	3	2	1	0
flags					bytes		

where

bytes: width of target locations in bytes. Either 1, 2 or 4

flags: control flags for command behavior.

Data Mask = bit 3: if set, only specific bits may be overwritten at target address (otherwise all bits may be overwritten)

Data Set = bit 4: if set, bits at the target address overwritten with this flag (otherwise it is ignored)

One or more target address and value/bitmask pairs can be specified. The same bytes and flags parameters apply to all locations in the command.

When successful, this command writes to each target address in accordance with the flags as follows:

**Table 8-28. Interpretation of Write Data Command Flags**

"Mask"	"Set"	Action	Interpretation
0	0	*address = val_msk	Write value
0	1	*address = val_msk	Write value
1	0	*address &= ~val_msk	Clear bitmask
1	1	*address  = val_msk	Set bitmask



**NOTE**

If any of the target addresses does not have the same alignment as the data width indicated in the parameter field, none of the values are written.

If any of the values is larger or any of the bitmasks is wider than permitted by the data width indicated in the parameter field, none of the values are written.

If any of the target addresses do not lie within an allowed region, none of the values are written. The list of allowable blocks and target addresses for the Chip are given below.

**Table 8-29. Valid DCD Address Ranges**

Address range	Start address	Last Address
IOMUX Control (IOMUXC) registers	0x020E0000	0x020E3FFF
CCM register set	0x020C4000	0x020C7FFF
MMDC register set	0x021B0000	0x021B7FFF
IRAM Free Space	0x00907000	0x00937FF0
EIM - Memory	0x08000000	0x0FFEFFFFF
EIM - Registers	0x021B8000	0x021BBFFF
DDR	0x10000000	0xFFFFFFFF

**8.6.2.2 Check Data Command**

The Check Data Command is used to test for a given -1, 2- or 4-byte bitmasks from a source address.

The Check Data Command is a big endian byte array with format shown in the table below.

**Table 8-30. Check Data Command Format**

Tag	Length	Parameter
	Address	
	Mask	
	[Count]	

where:

Tag: A single byte field set to 0xCF

Length: A two byte field in big endian format containing the length of the Check Data Command, in bytes, including the header

Address: source address to test

## Program image

Mask: bit mask to test

Count: optional poll count. If count is not specified this command will poll indefinitely until the exit condition is met. If count = 0, this command behaves as for NOP.

The Parameter field is a single byte divided into bitfields as follows:

**Table 8-31. Check Data Command Parameter field**

7	6	5	4	3	2	1	0
flags					bytes		

where

bytes: width of target locations in bytes. Either 1, 2 or 4

flags: control flags for command behavior.

Data Mask = bit 3: if set, only specific bits may be overwritten at target address (otherwise all bits may be overwritten)

Data Set = bit 4: if set, bits at the target address overwritten with this flag (otherwise it is ignored)

This command polls the source address until either the exit condition is satisfied, or the poll count is reached. The exit condition is determined by the flags as follows:

**Table 8-32. Interpretation of Check Data Command Flags**

"Mask"	"Set"	Action	Interpretation
0	0	(*address & mask) == 0	All bits clear
0	1	(*address & mask) == mask	All bits set
1	0	(*address & mask) != mask	Any bit clear
1	1	(*address & mask) != 0	Any bit set

### NOTE

If the source address does not have the same alignment as the data width indicated in the parameter field, the value is not read.

If the bitmask is wider than permitted by the data width indicated in the parameter field, the value is not read.

### 8.6.2.3 NOP Command

This command has no effect.

The format of NOP Command is a big endian four byte array as shown in the table below.

**Table 8-33. NOP Command Format**

Tag	Length	Undefined
-----	--------	-----------

where:

Tag: A single byte field set to 0xC0

Length: A two byte field in big endian containing the length of the NOP Command in bytes.  
Fixed to a value of 4.

Undefined: This byte is ignored and can be set to any value.

### 8.6.2.4 Unlock Command

The Unlock Command is used to prevent specific engine features being locked when exiting ROM.

The format of Unlock Command, again a big endian byte array, is shown in the table below.

**Table 8-34. Unlock Command Format**

Tag	Length	Eng
	Value	
	Value	
	...	
	Value	

where:

Tag: A single byte field set to 0xB2

Eng: Engine to be left unlocked, supported engines are SNVS and OCOTP.

Values: [optional] unlock values required by engine. Valid values are:

for SNVS: 0x01 (HAB\_SNVS\_UNLOCK\_LP\_SWR) - leave LP SW reset unlocked (leave LP\_SWR bit cleared in register SNVS\_HP command register) and  
0x02 (HAB\_SNVS\_UNLOCK\_ZMK\_WRITE) - leave zeroisable master key write unlocked (leave ZMK\_WSL bit cleared in register SNVS\_HP lock register).

for OCOTP:

0x01 (HAB\_OCOTP\_UNLOCK\_FIELD\_RETURN) - leave field return activation unlocked in OCOTP\_SW\_STICKY register,

0x02 (HAB\_OCOTP\_UNLOCK\_SRK\_REVOKE) - leave SRK revocation unlocked in OCOTP\_SW\_STICKY register,

0x04 (HAB\_OCOTP\_UNLOCK\_SCS) - leave lock bit cleared in OCOTP\_SCS register and

0x08 (HAB\_OCOTP\_UNLOCK\_JTAG) - leave HAB\_JDE bit cleared in OCOTP\_SCS register to unlock JTAG.

#### NOTE

This command may not be used in DCD structure if the SEC\_CONFIG is configured as closed.

## 8.7 Plugin Image

The ROM supports a limited number of boot devices. For using other devices as boot source (for example, Ethernet, CDROM, or USB), the supported boot device must be used (typically serial ROM) for firmware with the missing boot drivers. >Additionally plugin can customize supported boot drivers. It is more flexible when doing device initialization, such as condition judging, delay assertion, applying custom settings to boot device and memory system.

>In addition to standard images, the chip also supports plugin images. Plugin images return execution to the ROM whereas a standard image does not.

The boot ROM detects the image type using the plugin flag of the boot data structure (see [Boot Data Structure](#)). If the plugin flag is 1, then the ROM uses the image as a plugin function. The function must initialize the boot device and copy the program image to the final location. At the end the plugin function must return with the program image parameters. (See [High level boot sequence](#) for details about boot flow).

The boot ROM authenticates the plugin image prior to running the plugin function and then authenticates the program image.

The plugin function must follow the API described below:

```
typedef unsigned char (*) plugin_download_f(void **start, size_t *bytes, UINT32
*ivt_offset)
```

### ARGUMENTS PASSED:

- start - Image load address on exit.
- bytes - Image size on exit.
- ivt\_offset - Offset in bytes of the IVT from the image start address on exit.

### RETURN VALUE:

- 1 - on success
- 0 - on failure

## 8.8 Serial Downloader

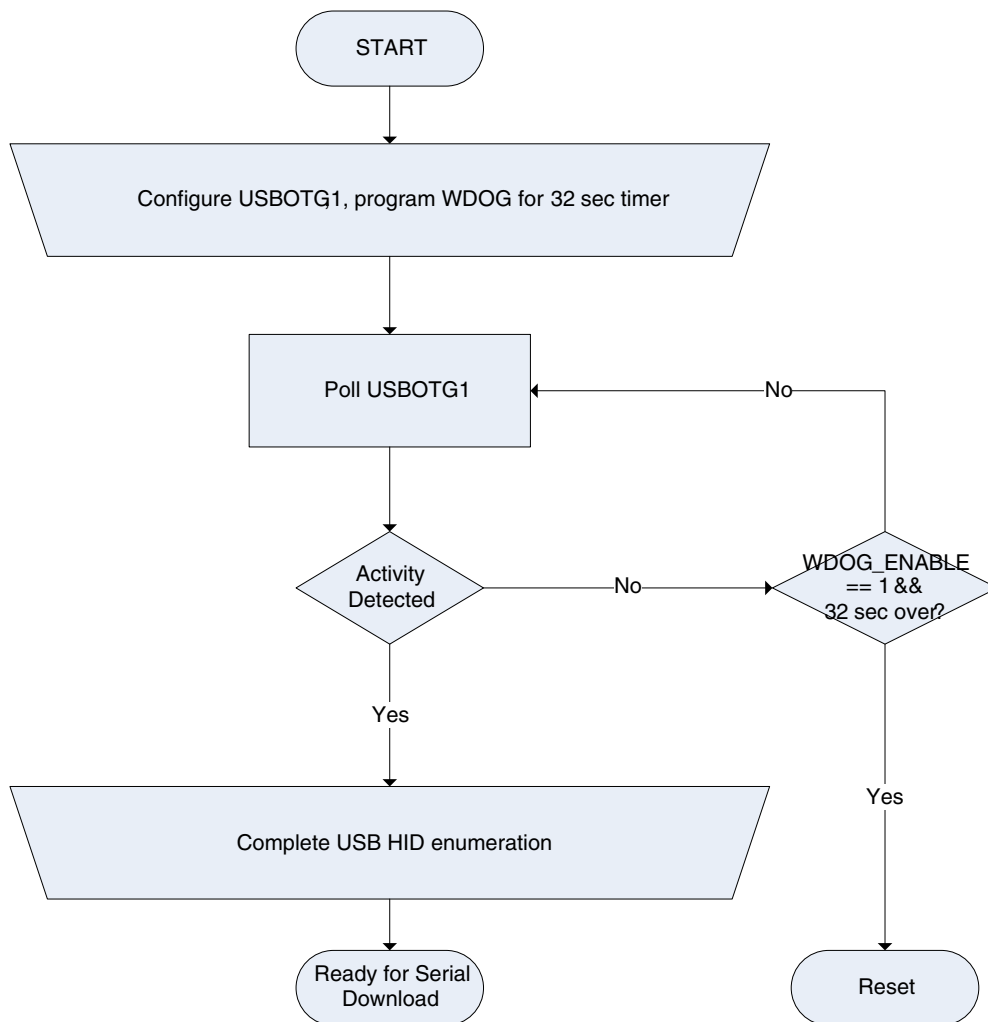
The Serial Downloader provides a means to download a Program Image to the chip over USB serial connection.

In this mode the ROM programs WDOG-1 for a 32-second time-out if WDOG\_ENABLE eFuse is 1, and then continuously polls for USB connection. If no activity is found on USB OTG1 and the watchdog timer expires, the ARM core is reset.

**NOTE**

The downloaded image must continue to service the watchdog timer to avoid an undesired reset from occurring.

The USB boot flow is shown in the figure below.



**Figure 8-16. USB Boot Flow**

### 8.8.1 USB

USB support is composed of the USBOH3 (USB OTG1 core controller, compliant with the USB 2.0 specification) and the USBPHY (HS USB transceiver).

The ROM supports the USB OTG port for boot purposes. The other USB ports on the chip are not supported for boot purposes.

The USB Driver is implemented as a USB HID class. A collection of 4 HID reports are used to implement SDP protocol for data transfers as described in [Table 8-35](#).

**Table 8-35. USB HID Reports**

Report ID (first byte)	Transfer Endpoint	Direction	Length	Description
1	control OUT	Host to device	17 bytes	SDP command from host to device
2	control OUT	Host to device	Up to 1025 bytes	Data associated with report 1 SDP command
3	interrupt	Device to host	5 bytes	HAB security configuration. Device sends 0x12343412 in closed mode and 0x56787856 in open mode.
4	interrupt	Device to host	Up to 65 bytes	Data in response to SDP command in report 1

### 8.8.1.1 USB Configuration Details

The USB OTG function device driver supports a high speed (HS for UTMI) non-stream mode with a maximal packet size of 512 B and a low-level USB OTG function.

The VID/PID and strings for USB device driver are listed in the table below.

**Table 8-36. VID/PID and Strings for USB Device Driver**

Descriptor	Value
VID	0x15A2 (Freescale vendor ID)
PID <sup>1</sup>	
String Descriptor1 (manufacturer)	Freescale Semiconductor, Inc.
String Descriptor2 (product)	S Blank SE Blank NS Blank
String Descriptor4	Freescale Flash
String Descriptor5	Freescale Flash

1. Allocation based on BPN (Before Part Number)

### 8.8.1.2 IOMUX Configuration for USB

The interface signals of the UTMI PHY are not configured in the IOMUX. The UTMI PHY interface uses dedicated contacts on the IC. See the Chip data sheet for details.

## 8.8.2 Serial Download protocol

The 16 byte SDP command from host to device is sent using HID report 1.

The table below describes 16 byte SDP command data structure:

**Table 8-37. 16 Byte SDP Command Data Structure**

BYTE Offset	Size	Name	Description
0	2	COMMAND TYPE	The following commands are supported for ROM: <ul style="list-style-type: none"> <li>• 0x0101 READ_REGISTER</li> <li>• 0x0202 WRITE_REGISTER</li> <li>• 0x0404 WRITE_FILE</li> <li>• 0x0505 ERROR_STATUS</li> <li>• 0x0A0A DCD_WRITE</li> <li>• 0x0B0B JUMP_ADDRESS</li> </ul>
2	4	ADDRESS	Only relevant for following commands: READ_REGISTER, WRITE_REGISTER, WRITE_FILE, DCD_WRITE, and JUMP_ADDRESS.  For READ_REGISTER and WRITE_REGISTER commands, this field is address to a register. For WRITE_FILE and JUMP_ADDRESS commands, this field is an address to internal or external memory address.
6	1	FORMAT	Format of access, 0x8 for 8-bit access, 0x10 for 16-bit and 0x20 for 32-bit access. Only relevant for READ_REGISTER and WRITE_REGISTER commands.
7	4	DATA COUNT	Size of data to read or write. Only relevant for WRITE_FILE, READ_REGISTER, WRITE_REGISTER and DCD_WRITE commands. For WRITE_FILE and DCD_WRITE commands DATA COUNT is in byte units.
11	4	DATA	Value to write. Only relevant for WRITE_REGISTER command.
15	1	RESERVED	Reserved

### 8.8.2.1 SDP Command

SDP commands are described in the following sections.

### 8.8.2.1.1 READ REGISTER

The transaction for command READ\_REGISTER consists of following reports: Report1 for command, Report3 for security configuration and Report4 for response or register value.

The register to read is specified in ADDRESS field of SDP command. First device sends Report3 with security configuration followed by Report4 with bytes read at given address. If count is greater than 64 then multiple reports with report id 4 are sent until entire data requested by host is sent. The STATUS is either 0x12343412 for closed parts and 0x56787856 for open or field return parts.

Report1, Command, Host to Device:

1	Valid values for READ_REGISTER COMMAND, ADDRESS, FORMAT, DATA_COUNT
---	---

ID 16 byte SDP Command

Report3, Response, Device to Host:

3	4 bytes indicating security configuration
---	---

ID 4 bytes status

Report4, Response, Device to Host: first response report

4	Register Value
---	----------------

ID 4 bytes of data containing register value. If number of bytes requested is less than 4 then remaining bytes should be ignored by host.

Multiple reports of report id 4 are sent until entire data requested is sent

Report4, Response, Device to Host: Last response report

4	Register Value
---	----------------

ID 64 bytes of data containing register value. If number of bytes requested is less than 64 then remaining bytes should be ignored by host.

### 8.8.2.1.2 WRITE REGISTER

The transaction for command WRITE\_REGISTER consists of the following reports: Report1 for command, Report3 for security configuration and Report4 for write status.



Host sends Report1 with WRITE\_REGISTER command. The register to write is specified in ADDRESS field of SDP command of Report1, with FORMAT field set to data type (number of bits to write 8, 16 or 32) and value to write in DATA field of SDP command. Device writes the DATA to register address and returns WRITE\_COMPLETE code using Report4 and security configuration using Report3 to complete the transaction.

Report1, Command, Host to Device:

1	Valid values for WRITE_REGISTER COMMAND, ADDRESS, FORMAT, DATA_COUNT and DATA
---	---

ID 16 byte SDP Command

Report3, Response, Device to Host:

3	4 bytes indicating security configuration
---	---

ID 4 bytes status

Report4, Response, Device to Host:

4	WRITE_COMPLETE (0x128A8A12) status
---	------------------------------------

ID 64 bytes data with first 4 bytes to indicate write is completed with code 0x128A8A12. On failure device will report HAB error status.

### 8.8.2.1.3 WRITE\_FILE

The transaction for command WRITE\_FILE consists of following reports: Report1 for command-phase, Report2 for data-phase, Report3 for hab mode and Report4 to indicate data received in full.

The size of each Report2 is limited to 1024 bytes (limitation of USB HID protocol) hence multiple Report2 packets will be sent by host in data phase until entire data is transferred to device. Once entire data (DATA\_COUNT bytes) is received then device sends report 3 with hab mode and report 4 with 0x88888888, indicating file download completed.

Report1, Host to Device:

1	Valid values for WRITE_FILE COMMAND, ADDRESS, DATA_COUNT
---	--

ID 16 byte SDP Command

=====Optional Begin=====

Host sends ERROR\_STATUS command to query if HAB rejected the address

===== Optional End=====

Report2, Host to Device:

2	File data
---	-----------

ID Max 1024 bytes data per report

Report2, Host to Device:

2	File data
---	-----------

ID Max 1024 bytes data per report

Report3, Device to Host:

3	4 bytes indicating security configuration
---	---

ID 4 bytes status

Report4, Response, Device to Host:

4	COMPLETE (0x88888888) status
---	------------------------------

ID 64 bytes data with first 4 bytes to indicate file download has completed with code 0x88888888. On failure device will report HAB error status.

#### 8.8.2.1.4 ERROR\_STATUS

The transaction for SDP command ERROR\_STATUS consists of three reports.

Report1 is used by host to send the command; device sends global error status in 4 bytes of Report4 after returning security configuration in Report3. When device receives ERROR\_STATUS command it will return global error status that is updated for each command. This command is useful to find out if last command resulted in device error or succeeded.

Report1, Command, Host to Device:

1	ERROR_STATUS COMMAND
---	----------------------

ID 16 byte SDP Command

Report3, Response, Device to Host:

3	4 bytes indicating security configuration
---	---

ID 4 bytes status

Report4, Response, Device to Host:

4	4 bytes Error status
---	----------------------

ID first 4 bytes status in 64 bytes report 4

### 8.8.2.1.5 DCD WRITE

The SDP command DCD\_WRITE is used by host to send multiple register writes in one shot. This command is provided to speed up the process of programming register writes such as to configure external RAM device.

The command goes with Report1 from host with COMMAND TYPE set to DCD\_WRITE, ADDRESS which is used for temporary location of DCD data and DATA\_COUNT to number of bytes sent in data out phase. In data phase host sends data for number of registers using Report2. Device completes the transaction with Report3 indicating security configuration and report 4 with WRITE\_COMPLETE code 0x12828212.

Report1, Command, Host to Device:

1	DCD_WRITE COMMAND, ADDRESS, DATA_COUNT
---	--

ID 16 byte SDP Command

Report2, Data, Host to Device:

2	DCD binary data
---	-----------------

ID Max 1024 bytes per report

Report3, Response, Device to Host:

3	4 bytes indicating security configuration
---	---

ID 4 bytes status

Report4, Response, Device to Host:

4	WRITE_COMPLETE (0x128A8A12) status
---	------------------------------------

ID 64 bytes report with first 4 bytes to indicate write is completed with code 0x128A8A12. On failure device will report HAB error status.

See [Device Configuration Data \(DCD\)](#) for DCD format description.

### 8.8.2.1.6 JUMP ADDRESS

The SDP command JUMP\_ADDRESS will be the last command host can send to the device, after this command device will jump to the address specified in the ADDRESS field of SDP command and start executing.

This command should typically follow after WRITE\_FILE command. The command is sent by host in command-phase of transaction using Report1, there is no data phase for this command but device send status report3 to complete the transaction. And if HAB authentication fails then it will also send report 4 with HAB error status.

Report1, Command, Host to Device:

1	JUMP_ADDRESS COMMAND, ADDRESS
---	-------------------------------

ID 16 byte SDP Command

Report3, Response, Device to Host:

3	4 bytes indicating security configuration
---	---

ID 4 bytes status

This report is sent by device only in case of an error jumping to the given address, device reports error in Report4, Response, Device to Host:

4	4 bytes HAB error status
---	--------------------------

ID 4 bytes status, 64 bytes report length

## 8.9 Recovery Devices

The Chip supports recovery devices. If primary boot device fails, boot ROM will try to boot from recovery device using one of I2C or ECSPI ports.

For enabling recovery device BOOT\_CFG4[6] fuse must be set. Additionally Serial EEPROM fuses must be set as described in [Serial ROM through SPI and I2C](#).

## 8.10 USB Low Power Boot

ROM supports USB Low Power Boot. This feature enables a device with dead or weak battery to power up and boot if the device is connected to a USB upstream port, no matter the upstream port is a USB charger or USB host/hub.

If a USB dedicated charger or host/hub charger are connected, as soon as the device is connected to the upstream port, a stable current (Max.1.5A) can be supplied by charger. If USB host/hub are connected, the maximal 100mA current is supplied to the device, the device should be able to power up to boot the image with less than 100mA.

If LPB\_BOOT fuses are blown, the Chip will check if there is low power condition via GPIO\_3 pad. If there is low power boot condition USB charger detection will be activated. If there is no USB charger, ROM will initialize USB as device and apply division factors on ARM, DDR, AXI and AHB root clocks based on LPB\_BOOT fuses value (see the table below). Polarity of low power boot condition on GPIO\_3 pad is set by BT\_LPB\_POLARITY fuse (see the figure below).

**Table 8-60. USB Low Power Boot Frequencies**

LPB_BOOT	Boot Frequencies=0	Boot Frequencies=1
00	ARM_CLK_ROOT=792MHz MMDC_CH0_AXI_CLK_ROOT=528MHz MMDC_CH1_AXI_CLK_ROOT=528MHz AXI_CLK_ROOT=264MHz AHB_CLK_ROOT=132MHz	ARM_CLK_ROOT=396MHz MMDC_CH0_AXI_CLK_ROOT=352MHz MMDC_CH1_AXI_CLK_ROOT=352MHz AXI_CLK_ROOT=176MHz AHB_CLK_ROOT=88MHz
01	ARM_CLK_ROOT=792MHz MMDC_CH0_AXI_CLK_ROOT=528MHz MMDC_CH1_AXI_CLK_ROOT=528MHz AXI_CLK_ROOT=264MHz AHB_CLK_ROOT=132MHz	ARM_CLK_ROOT=396MHz MMDC_CH0_AXI_CLK_ROOT=352MHz MMDC_CH1_AXI_CLK_ROOT=352MHz AXI_CLK_ROOT=176MHz AHB_CLK_ROOT=88MHz
10	ARM_CLK_ROOT=396MHz MMDC_CH0_AXI_CLK_ROOT=264MHz MMDC_CH1_AXI_CLK_ROOT=264MHz AXI_CLK_ROOT=132MHz AHB_CLK_ROOT=66MHz	ARM_CLK_ROOT=264MHz MMDC_CH0_AXI_CLK_ROOT=176MHz MMDC_CH1_AXI_CLK_ROOT=176MHz AXI_CLK_ROOT=88MHz AHB_CLK_ROOT=44MHz
11	ARM_CLK_ROOT=264MHz MMDC_CH0_AXI_CLK_ROOT=132MHz MMDC_CH1_AXI_CLK_ROOT=132MHz AXI_CLK_ROOT=66MHz AHB_CLK_ROOT=66MHz	ARM_CLK_ROOT=132MHz MMDC_CH0_AXI_CLK_ROOT=88MHz MMDC_CH1_AXI_CLK_ROOT=88MHz AXI_CLK_ROOT=44MHz AHB_CLK_ROOT=44MHz

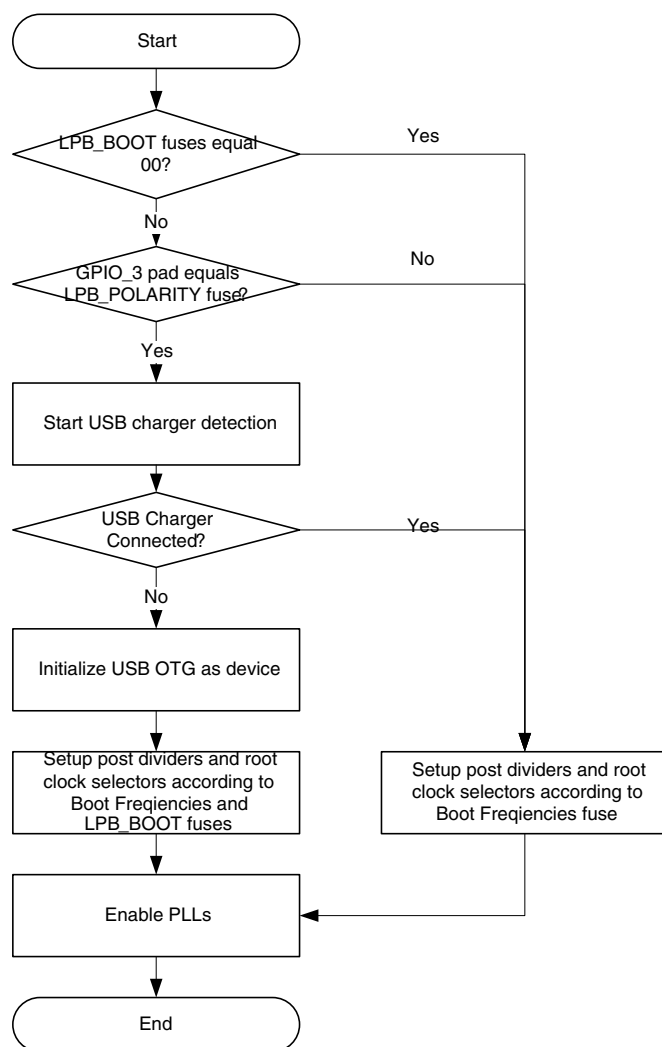


Figure 8-17. USB Low Power Boot Flow

## 8.11 SD/MMC Manufacture Mode

When internal boot and recover boot (if enabled) failed, boot will go to SD/MMC manufacture mode before serial download mode. In manufacture mode, one bit bus width is used despite of the fuse setting.

In manufacture mode, SD or MMC card will be scanned on uSDHC1 only. If card is detected and valid boot image is found in card, then boot image will be loaded then executed. Pad of SD1\_CD is used to detect whether card is inserted.

By default, SD/MMC manufacture mode is enabled, blow the fuse of DISABLE\_SDMMC\_MFG to disable it.

### NOTE

Secondary boot is not supported on SD/MMC manufacture mode.

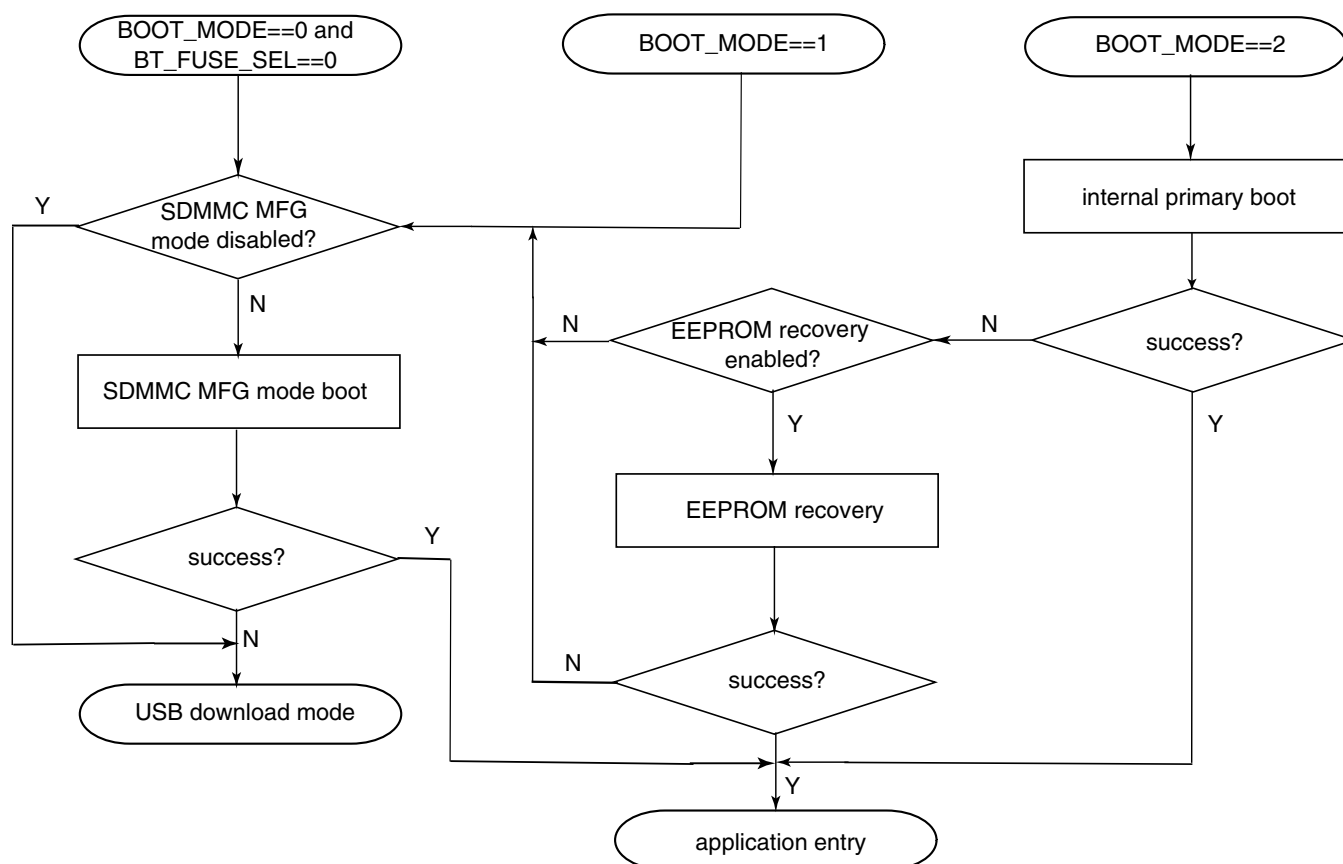
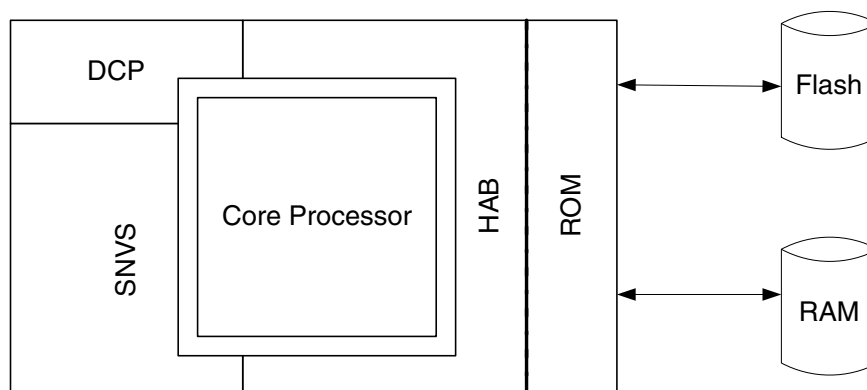


Figure 8-18. SD/MMC Manufacture boot flow

## 8.12 High Assurance Boot (HAB)

The High Assurance Boot (HAB) component of the ROM protects against the potential threat of attackers modifying areas of code or data in programmable memory to make it behave in an incorrect manner. The HAB also prevents attempts to gain access to features which should not be available.

The integration of the HAB feature with the ROM code ensures that Chip does not enter an operational state if the existing hardware security blocks have detected a condition that may be a security threat or areas of memory deemed to be important have been modified. The HAB uses RSA digital signatures to enforce these policies.



**Figure 8-19. Secure Boot Components**

The figure above illustrates the components used during a secure boot using HAB. The HAB interfaces with the SNVS to ensure the system security state is as expected. The HAB also makes use of DCP hardware block to accelerate SHA-256 message digest operations performed during signature verifications. The HAB also includes a software implementation of SHA-256 for cases where a hardware accelerator cannot be used. The RSA key sizes supported are 1024, 2048 and 3072 bits. The RSA signature verification operations are performed by a software implementation contained in the HAB library. The main features supported by HAB are:

- X.509 Public key certificate support
- CMS signature format support

#### **NOTE**

Freescall provides a reference Code Signing Tool (CST) for key generation, certificate generation and code signing for use with the HAB library. The CST can be found by searching for "IMX\_CST\_TOOL" at <http://www.freescall.com>.

#### **NOTE**

For further details on making use of the secure boot feature using HAB contact your local Freescall representative.



### 8.12.1 ROM Vector Table Addresses

For devices that perform a secure boot, the HAB library may be called by boot stages that execute after ROM code. The RVT table contains the pointers to the HAB API functions and is located at 0x00000094.

#### NOTE

For additional information on secure boot including the HAB API, contact your local Freescale representative.



## Chapter 9

# Multimedia

### 9.1 Display and graphics subsystem

The chip display and graphics subsystem consists of the dedicated modules found here.

- EPDC (electrophoretic display controller): display controller for E-INK EPD panel
- SPDC (SiPix display controller): display controller for SiPix EPD panel
- LCDIF (LCD interface): 24-bit parallel RGB LCD interface
- PXP pixel pipeline: pixel/image processing engine for EPD and LCD display;
- Two graphic processing units:
  - GPU2D: acceleration the generation of 2D graphics (BitBLT).
  - GPUVG: acceleration of vector graphics (OpenVG).
- CSI (camera sensor interface): up to 16-bit parallel interface for image sensor

The following figure shows the high level integration scheme of the chip display and graphics system.

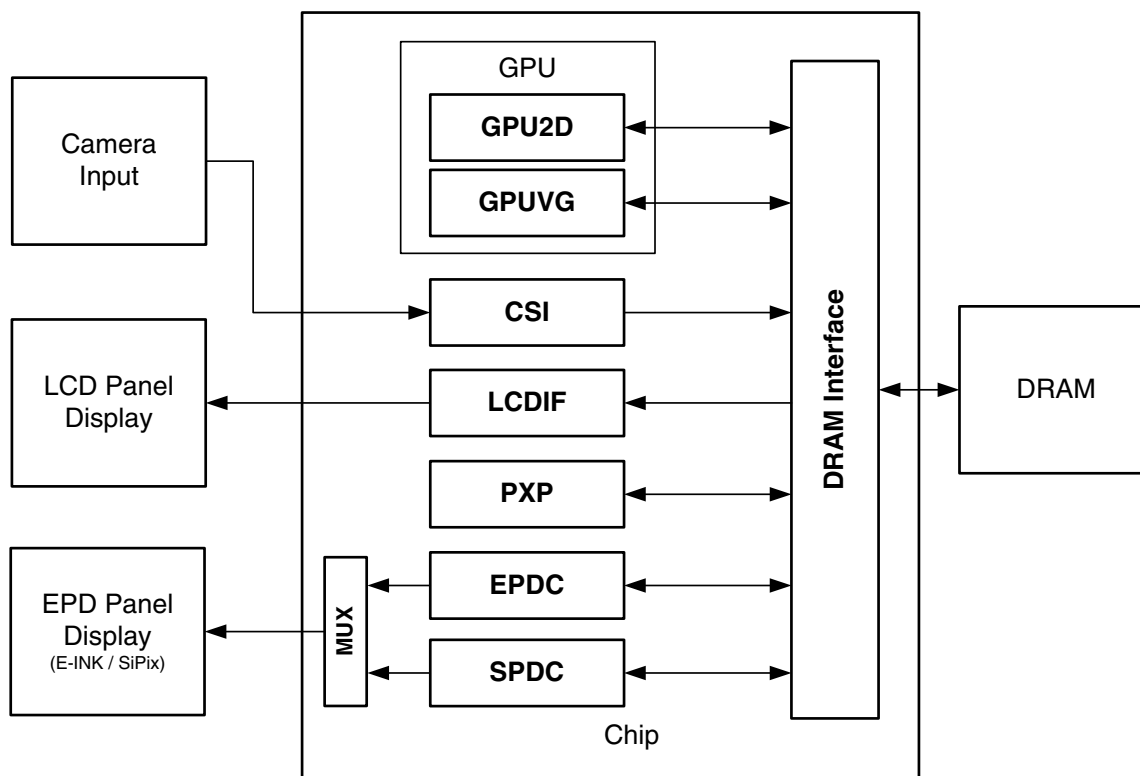


Figure 9-1. Chip display and graphics subsystem

### 9.1.1 Electrophoretic Display Controller

The EPDC is a feature-rich, low power, and high-performance direct-drive active matrix EPD controller. It is specifically designed to drive E•INK™ EPD panels supporting a wide variety of TFT backplanes. The goal of the EPDC is to provide an efficient SoC integration of this functionality for e-paper applications, allowing a significant BOM cost saving over an external solution whilst reaching much higher levels of performance and lower power. The EPDC module is defined in the context of an optimized HW/SW partitioning and works in conjunction with the ePXP IP module to form a complete display processing solution.

The key features of the EPDC are as follows:

- Supports direct-driver for E-Ink EPD panels, with up to 2048x1536 resolution at 106 Hz refresh rate (or 4096x4096 resolution at 20 Hz refresh rate)
- Supports up to 64 LUT for concurrent update
- Supports automatic waveform selection based on both input image and current panel content
- Supports collision detection
- Supports both PVI panels and LGD GIP panels.

### 9.1.2 SiPix Display Controller

SiPix Display Controller is used to drive EPD panels based on SiPix technology. The main feature of this controller includes:

- Supports direct driver for SiPix EPD panels with up to 1600x1200.
- Support 8-bit / 16-bit panel interface.

### 9.1.3 PiXel Pipeline

The pixel pipeline is used to perform image processing on image/video buffers before sending to an LCD display or EPD display. It supports basic image operations including resizing, overlay, rotation, as well as CSC and CFA operation used for EPD display.

The main features of PXP include:

- Multiple input/output format support, including YUV/RGB/Grayscale
- Supports both RGB/YUV scaling
- Supports overlay with Alpha blending
- RGB656/RGB444 to RGBW4444 conversion with LUT for color EPD panel

### 9.1.4 LCD Interface

The LCDIF is a general purpose display controller that is used to drive a wide range of display devices. These displays can vary in size and capability. Many of these displays have had an asynchronous parallel MPU interface for command and data transfer to an integrated frame buffer. There are other popular displays that support moving pictures and require the RGB interface mode (called DOTCLK interface in this document) or the VSYNC mode for high-speed data transfers. In addition to these displays, it is also common to provide support for digital video encoders that accept ITU-R BT.656 format 4:2:2 YCbCr digital component video and convert it to analog TV signals. The LCDIF block supports these different interfaces by providing fully programmable functionality.

The block has several major features:

- Bus master interface to source frame buffer data for display refresh and a DMA interface to manage input data transfers from the LCD requiring minimal CPU overhead.
- 8/16/18/24/32 bit LCD data bus support available depending on I/O mux options.

- Programmable timing and parameters for MPU, VSYNC and DOTCLK LCD interfaces to support a wide variety of displays.
- ITU-R BT.656 mode (called Digital Video Interface or DVI mode here) including progressive-to-interlace feature and RGB to YCbCr 4:2:2 color space conversion to support 525/60 and 625/50 operation

### **9.1.5 CMOS Sensor Interface**

The CSI enables the chip to connect directly to external CMOS image sensors. CMOS image sensors are separated into two classes, dumb and smart. Dumb sensors are those that support only traditional sensor timing (Vertical SYNC and Horizontal SYNC) and output only Bayer and statistics data, while smart sensors support CCIR656 video decoder formats and perform additional processing of the image (for example, image compression, image pre-filtering, and various data output formats).

The capabilities of the CSI include:

- Configurable interface logic to support most commonly available CMOS sensors.
- Support for CCIR656 video interface as well as traditional sensor interface.
- 8-bit data port for YCC, YUV, or RGB data input.
- 8-bit/10-bit/16-bit data port for Bayer data input.
- Full control of 8-bit/pixel, 10-bit/pixel or 16-bit/pixel data format to 32-bit receive FIFO packing.
- 128 × 32 FIFO to store received image pixel data. Receive FIFO overrun protection mechanism.
- Embedded DMA controllers to transfer data from receive FIFO or statistic FIFO through AHB bus.
- Support 2D DMA transfer from the receive FIFO to the frame buffers in the external memory.
- Support double buffering two frames in the external memory.
- Single interrupt source to interrupt controller from maskable interrupt sources: Start of Frame, End of Frame, Change of Field, FIFO full, FIFO overrun, DMA transfer done, CCIR error and AHB bus response error.
- Configurable master clock frequency output to sensor.
- Statistic data generation for Auto Exposure (AE) and Auto White Balance (AWB) control of the camera (only for Bayer data and 8-bit/pixel format).

### **9.1.6 2D Graphics Processing Unit (GPU2Dv2)**

### 9.1.6.1 2D feature summary

GPU2D has the following features:

- Bit BLT & stretch BLT
- Rectangle fill and clear
- Line drawing
- High performance stretch and shrink
- Mono expansion for text rendering
- ROP2, ROP3, and ROP4
- Alpha blending including Java 2 Porter-Duff compositing blending rules
- Support rendering size of 32Kx32K
- 90/180/270 degree rotation
- Transparency by monochrome mask, chroma key, or pattern mask
- Color space conversion between YUV and RGB
- High quality image scaling, using up to 9x9 separable filter
- Bit-Blit Formats
  - A1R5G5B5 (source/destination)
  - A4R4G4B4 (source/destination)
  - X1R5G5B5 (source/destination)
  - X4R4G4B4 (source/destination)
  - R5G6B5 (source/destination)
  - X8R8G8B8 (source/destination)
  - A8R8G8B8 (source/destination)
  - 8-bit color index (source only)
  - A8 (source/destination)
  - 1-bit monochrome (source only)
- Filter Blit Formats
  - A1R5G5B5 (source/destination)
  - A4R4G4B4 (source/destination)
  - A8R8G8B8 (source/destination)
  - R5G6B5 (source/destination)
  - X1R5G5B5 (source/destination)
  - X4R4G4B4 (source/destination)
  - X8R8G8B8 (source/destination)
  - YUV (source only):
    - NV12 (4:2:0, 2 planes)
    - NV16 (4:2:2, 2 planes)
    - UYVY (4:2:2, interleave)

- YUY2 (4:2:2, interleave)
- YV12 (4:2:0, 3 planes)
- 8-bit color index(source only)

### 9.1.6.2 2D Performance

- Geometry Rate: 26.6M Triangles/sec
- Pixel Rate: 256M pixels/sec

### 9.1.6.3 2D Software

API / Driver Support

- GDI/DirectDraw
- DirectFB
- X11 EXA

Operating Systems

- Windows CE
- Linux Embedded and X11
- Android

## 9.1.7 Vector Graphics Processing Unit (GPUVGv2)

### 9.1.7.1 Vector Graphics Features

Featureset Overview:

- Real-time hardware curve tessellation of lines, quadratic and cubic Bezier curves
- 16x Line Anti-aliasing
- OpenVG 1.1 support
- Vector Drawing:
  - Coordinate Systems and Transformations
  - Viewport Clipping, Scissoring and Alpha Masking
  - Paths and stroke generation
  - Image interpolation
  - Image Filters
  - Paint (gradient and pattern)
  - Blending



### 9.1.7.2 Vector Graphics Performance

- GPU-VG @ 533MHz
- Full OpenVG 1.1 Khronos Conformance
- 533 M pixels / sec raw performance (1 pixel / clock)

### 9.1.7.3 Vector Graphics Software

API / Driver Support

- OpenVG 1.1

Operating Systems

- Windows CE
- Linux Embedded and X11

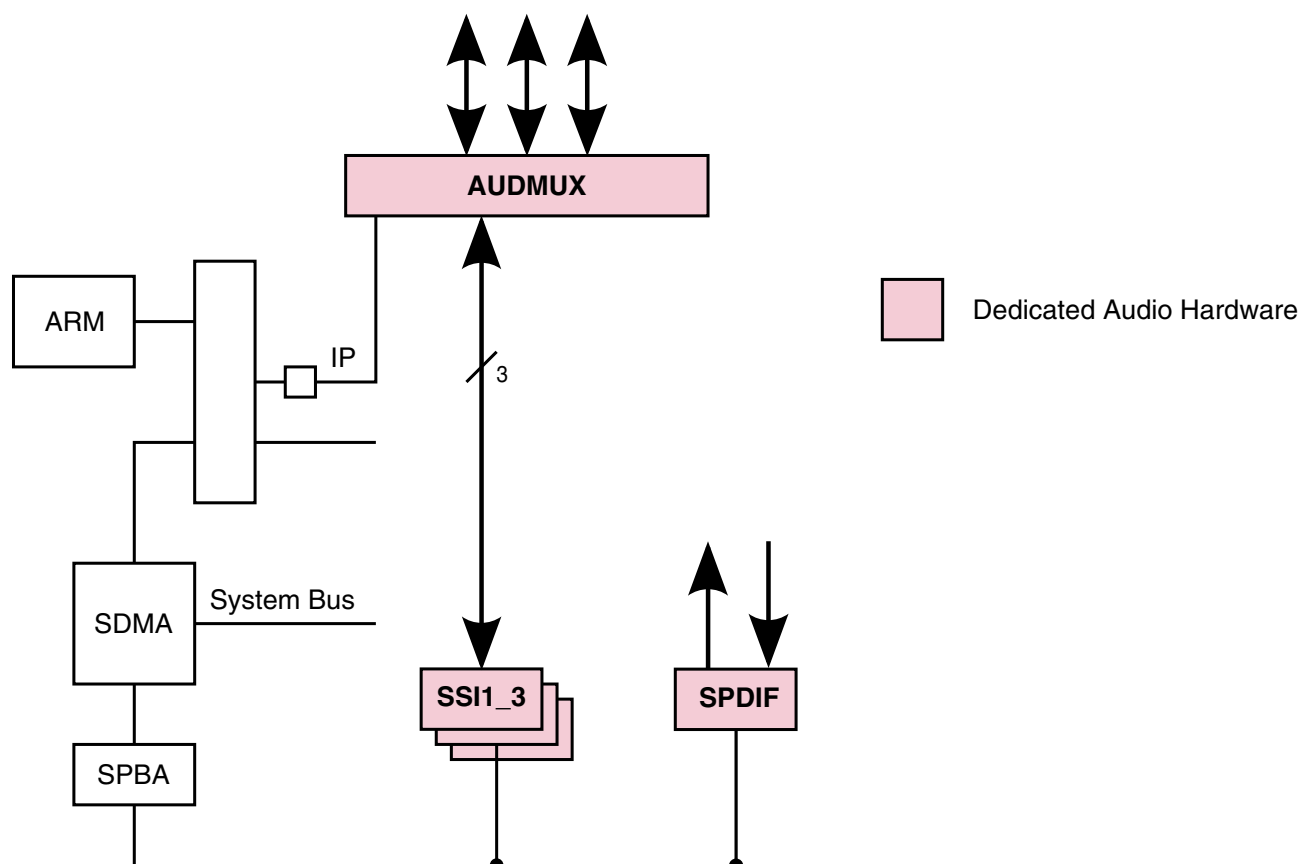
## 9.2 Audio subsystem

The audio subsystem consists of the following modules: SSI-1, SSI-2, SSI-3, AUDMUX, and SPDIF. In addition, the IOMUX must be appropriately configured to get signals in and out of the chip.

[Audio subsystem module overview](#) provides an overview of each of the audio subsystem component modules, followed by a module-specific section.

### 9.2.1 Audio subsystem module overview

The following figure shows a high level block diagram of the audio subsystem.



**Figure 9-2. Audio subsystem block diagram**

SSI1–3 are synchronous serial interfaces used to transfer audio data. SSI1–3 are on the shared peripheral bus. Instead of connecting to the IOMUX directly, their serial lines connect to the digital audio mux (AUDMUX).

AUDMUX handles data and clock signal routings to a particular SSI unit (SSI1, SS2, or SSI3) because SSI has no direct contact to the pin. AUDMUX routes audio data (and even splices together multiple time-multiplexed audio streams) but does not decode or process audio data itself. The ARM controls AUDMUX, but AUDMUX can route data even when the ARM is in a low-power mode.

The SPDIF (Sony/Philips digital interface) audio module is a stereo transceiver that allows the processor to receive and transmit digital audio over it. The SPDIF receiver section includes a frequency measurement block that allows the precise measurement of an incoming sampling frequency. A recovered clock is provided by the SPDIF receiver section and may be used to drive both internal and external components in the system. SPDIF is connected to the shared peripheral bus.

## 9.2.2 Synchronous Serial Interface (SSI)

The Synchronous Serial Interface (SSI) is a full-duplex serial port that allows communication with external devices using a variety of serial protocols. The SSI supports a wide variety of protocols (SSI normal, SSI network, I2S, and AC-97), bit depths (up to 24 bits per word), and clock/frame sync options.

The SSI has two pairs of 15x32 FIFOs and hardware support for an external DMA controller in order to minimize its impact on system performance. The second pair of FIFOs provides hardware interleaving of a second audio stream which reduces CPU overhead in use cases where two timeslots are being used simultaneously.

The three SSIs may support three audio streams (possibly at different sample rates) simultaneously. SSI1, SSI2 and SSI3 are located on the Shared Peripheral Bus. Since the SDMA can directly access SSI1...SSI3 (being on the Shared Peripheral Bus), they can be used for high-bandwidth data transfers in order to optimize bus bandwidth consumption.

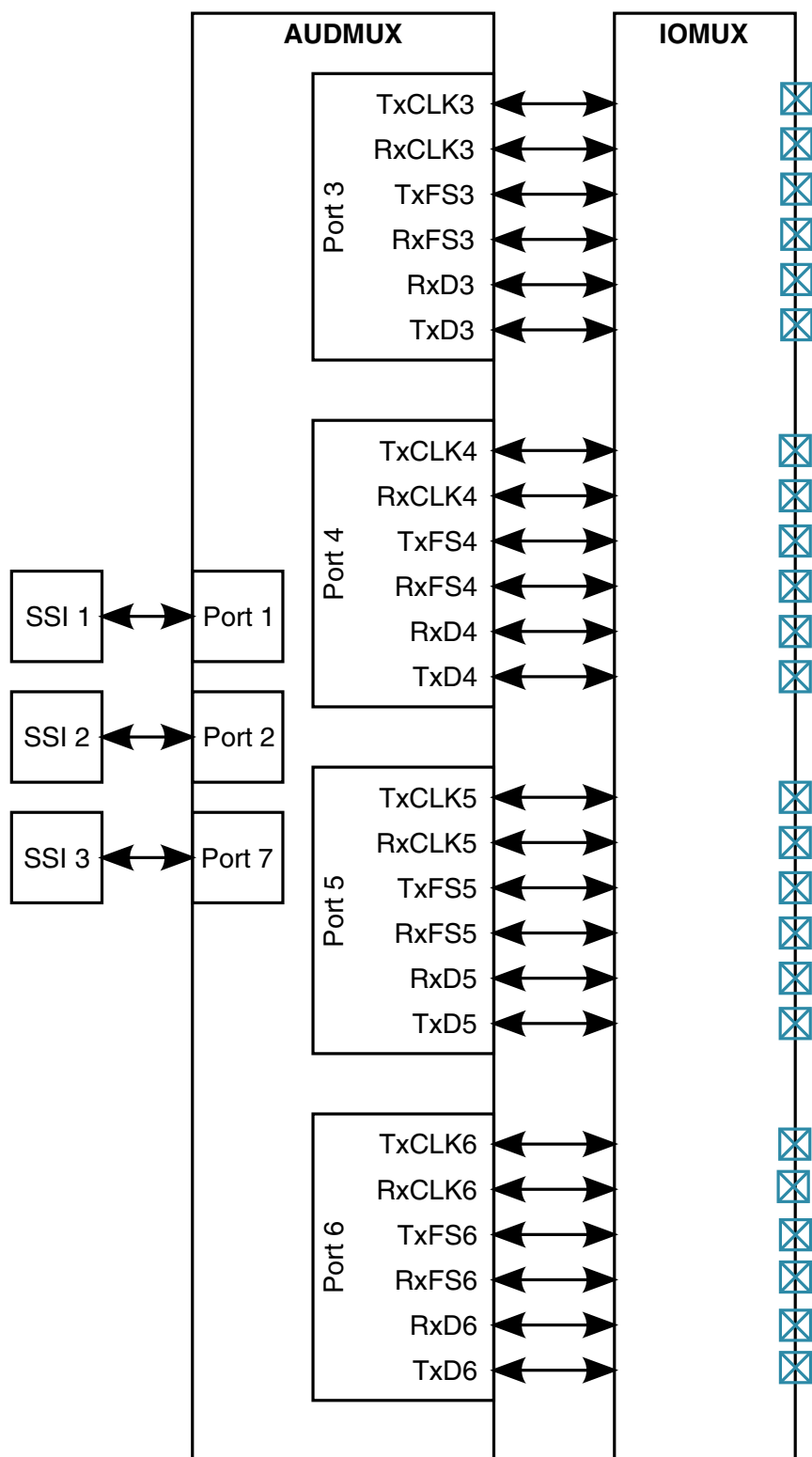
## 9.2.3 Digital Audio MUX (AUDMUX)

The Digital Audio Mux (AUDMUX) provides a programmable interconnect fabric for voice, audio, and synchronous data routing between host serial interfaces, such as SSI, and peripheral serial interfaces—that is, audio and voice codecs.

The AUDMUX includes two types of interfaces. Internal ports connect to the processor serial interfaces, and External ports connect to off-chip audio devices. A desired connectivity is achieved by configuring the appropriate host and peripheral ports.

The AUDMUX provides flexible, programmable routing of the on-chip serial interfaces to and from off-chip audio devices. The AUDMUX routes audio data (and even splices together multiple time-multiplexed audio streams) but does not decode or process audio data itself.

The following figure illustrates how the AUDMUX is connected in the system.



☒ - IO PAD (more than one PAD is available for each signal, only one option shown above).

**Figure 9-3. AUDMUX System Block Diagram**  
i.MX 6SoloLife Applications Processor Reference Manual, Rev. 1, 04/2013

## 9.2.4 Sony/Philips Digital Interface (SPDIF)

The Sony/Philips Digital Interface (SPDIF) module is a stereo that allows the processor transmit digital audio over it using the IEC60958 standard, consumer format. i.MX 6SoloLite provides one SPDIF transmitter with one output.

The SPDIF allows the handling of both SPDIF channel status (CS) and User (U) data.

For the SPDIF transmitter, the audio data is provided by the processor via the SPDIFTxLeft and SPDIFTxRight registers, and the data is stored in two 16-word-deep FIFOs, one for the right channel, the other for the left channel. The FIFOs support programmable watermark levels so that FIFO Empty service request can be triggered when the combined number of empty data words locations in both FIFOs is 8, 16, 24 or 32 words. It is recommended to program the watermark level to trigger a FIFO Empty service request when 16 word locations are empty. For optimal performance when servicing the FIFO Empty service request, the FIFOs should be written alternately, starting with the left channel FIFO. The Channel Status bits are also provided via the corresponding registers. The SPDIF transmitter generates an SPDIF output bitstream in the biphase mark format (IEC 60958), which consists of audio data, channel status and user bits.

The data handled by the SPDIF module is 24-bit wide. The 24-bit SPDIF data is aligned in the 24 least significant bits of the 32-bit shared peripheral bus data word. The 8 most significant bits of the 32-bit word are ignored by the SPDIF Transmitter when data is being stored in the Transmit FIFOs from the peripheral bus. The 8 most significant bits of the 32-bit word are zeroed by the SPDIF Receiver module when the data is being read from the Receiver FIFOs to the peripheral bus.

Note that 16-bit data is left-aligned in the 24-bit word format of the SPDIF. When 16-bit data is to be transmitted, the 32-bit word to be written to the SPDIF Transmit FIFOs should be created as follows: the 16-bit data should be located in the middle two bytes of the 32-bit data word and the 8 bits of the LSB must be set to zero, while the 8 bits of the MSB will be ignored.

The SPDIF Transmit clock is generated by the SPDIF internal clock generator module and the clock sources are from outside of the SPDIF block. The , SSI's clock sources should provide a clock that is at least  $64 \times F_s$ , where  $F_s$  is the sampling frequency. The external clock source should provide at least  $128 \times F_s$ . Clocks of higher frequency may be provided as long as the multiplication factor is a power of 2 (for example,  $128x$ ,  $256x$  or  $512x$ ). Also, clock frequency precision of 100ppm or better should be provided.

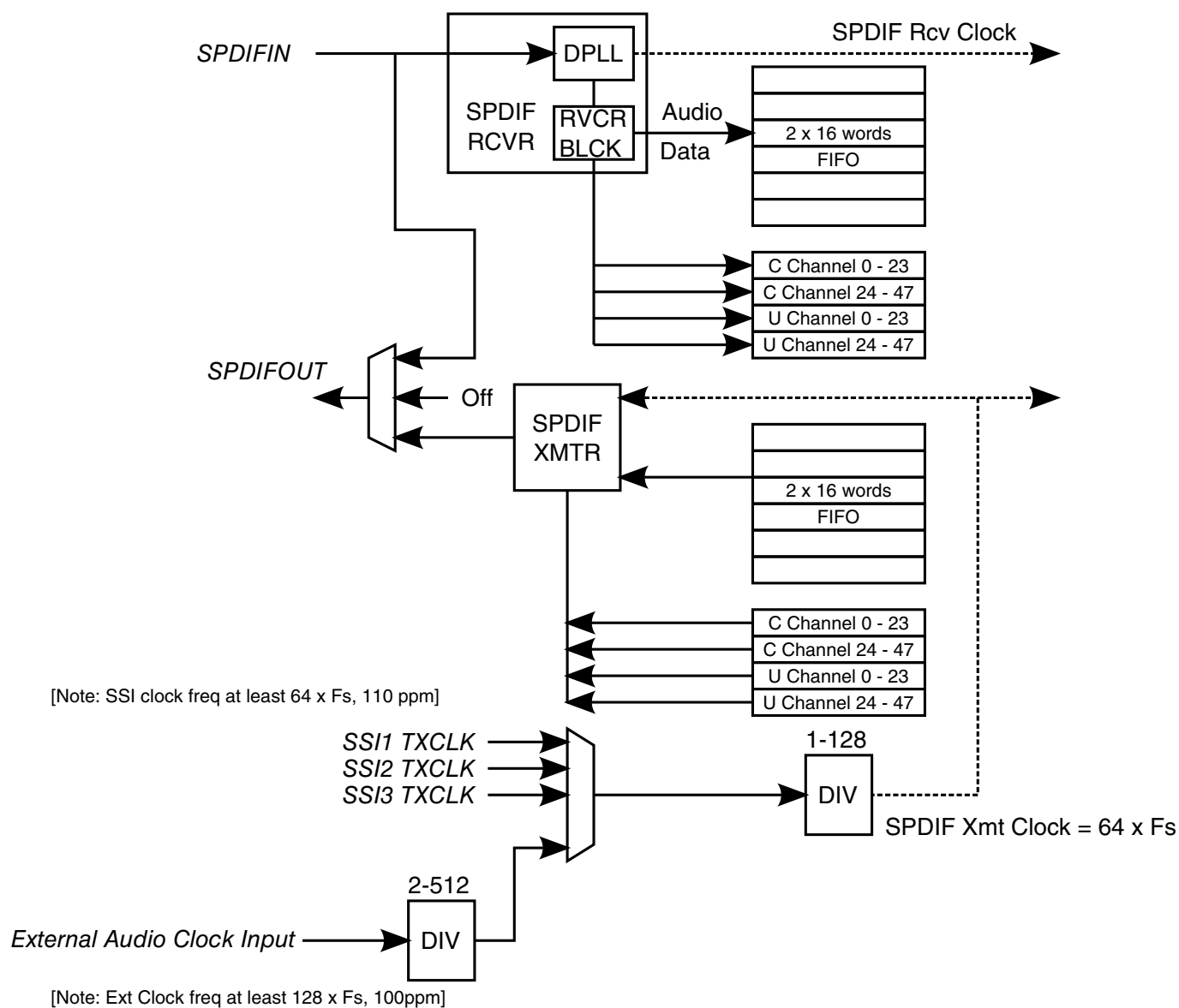


Figure 9-4. SPDIF Transceiver Clock Diagram

# Chapter 10

## Clock and Power Management

### 10.1 Introduction

This chapter describes the Clock and Power Management architecture of the SoC.

The device targets many applications where low power consumption, long battery life, always-on and instant-on capabilities, and no need for active cooling are paramount. For this reason, the design constantly focuses on reducing current consumption as much as possible, while simultaneously enabling the maximum level of peak performance and a balanced level of sustained performance for target applications. To achieve this, the architecture uses a wide range of power-management techniques and their combinations for maximum system design flexibility:

This introduction contains information about:

- Structural components of the power and clock management systems of the device
- Power, clock and thermal management techniques supported by the device

All the numerical values are typical or examples, for accurate values one should use the datasheet.

### 10.2 Device Power Management Architecture Components

To provide a clean and versatile architecture supporting a wide range of power-management techniques, the Clock and Power rails are considered managed resources.

For each rail, two levels of management are defined: the first level is centralized or SoC-level resource management, and the second is a local or "module level" resource management.

The high level architectural view of clock, power and thermal management system of device is presented on a figure below.

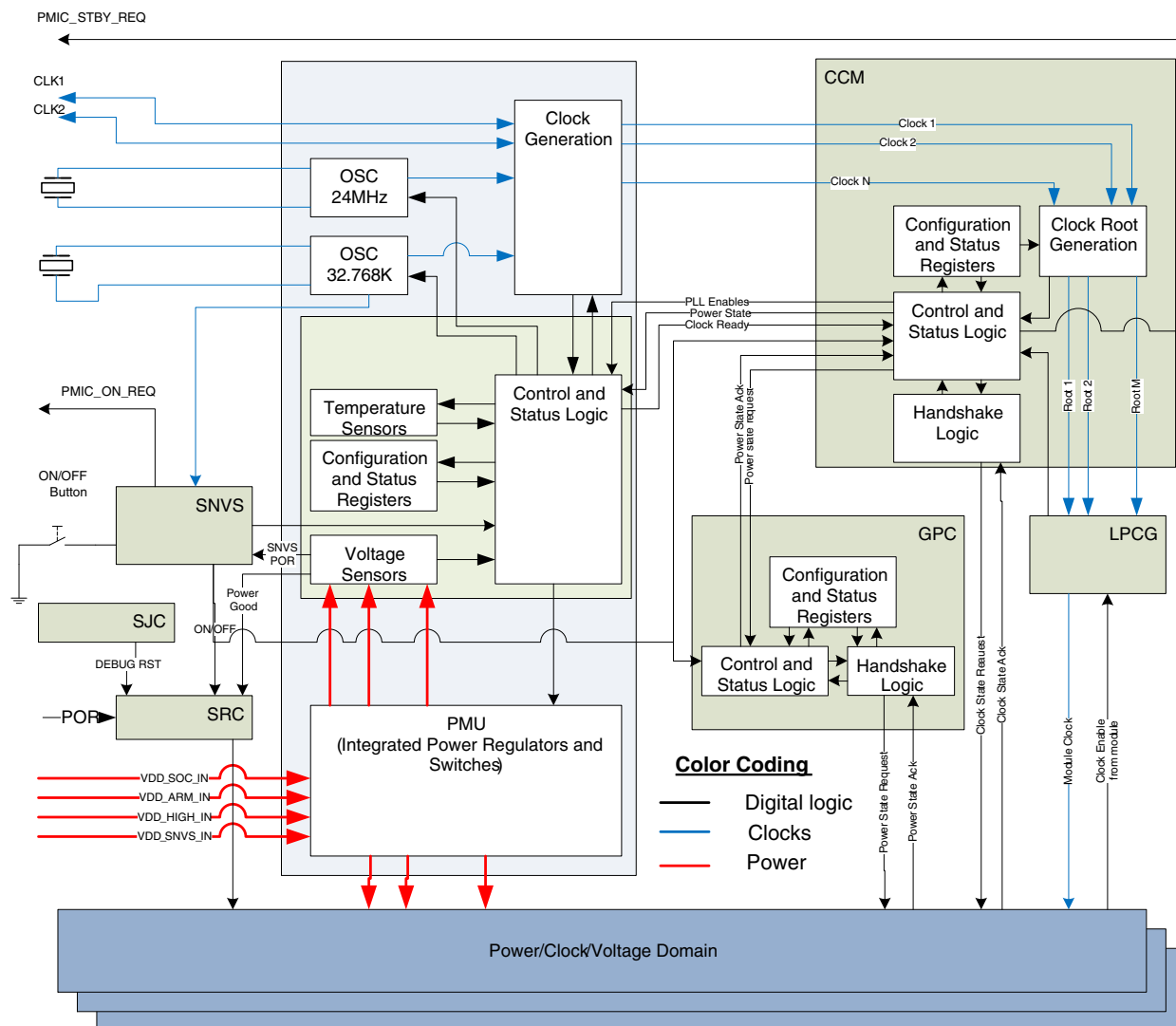


Figure 10-1. Power and clock management framework

## 10.2.1 Centralized components of clock generation and management

Centralized components of clock generation and management sub-system are implemented in the following blocks:

- **CCM (Clock Control Module):** The CCM module provides control for primary (source level) and secondary (root level) clock generation, division, distribution, synchronization, and coarse-level gating.



- See [Clock Controller Module \(CCM\)](#) for information on the CCM architecture, functional description and programming model.
- LPCG (Low Power Clock Gating): This module distributes the clocks to all blocks in the SoC and handles block level software-controllable and automated clock gating. See [Clock Controller Module \(CCM\)](#) for information on the LPCG architecture and functional description.

### 10.2.2 Centralized components of power generation, distribution and management

Centralized components of power generation, distribution and management sub-system are implemented in the following blocks:

- PMU (Integrated Power Management Unit). See [Power Management Unit \(PMU\)](#) for information on the PMU architecture, functional description and programming model.
- GPC (General Power Controller). See [General Power Controller \(GPC\)](#) for information on the GPC architecture, functional description and programming model.

### 10.2.3 Reset generation and distribution system

Power and clock management are accompanied with an appropriate reset generation and distribution system, centralized functions of which are implemented [System Reset Controller \(SRC\)](#). See [General Power Controller \(GPC\)](#) for information on the GPC architecture, functional description and programming model.

### 10.2.4 Power and clock management framework

Together, the modules listed above provide enhanced power-management features with the centralized control for the clock, reset, and power-management signals on the SoC.

The centralized management defines the minimal managed components of the power-management architecture. These components are called the clock, power, and voltage domains.

#### NOTE

A domain is a group of modules or functional blocks that share a common resource entity (for example, common clock root, common power source, or a common power switch). The software component managing shared resources should take

into account the joint constraints of all the modules belonging to that resource domain.

## **10.3 Clock Management**

### **10.3.1 Centralized components of clock management system**

The clock generation and management system is built around the CCM and LPCG blocks.

A high level block diagram of the clock management system in the SoC environment is shown in figure below.

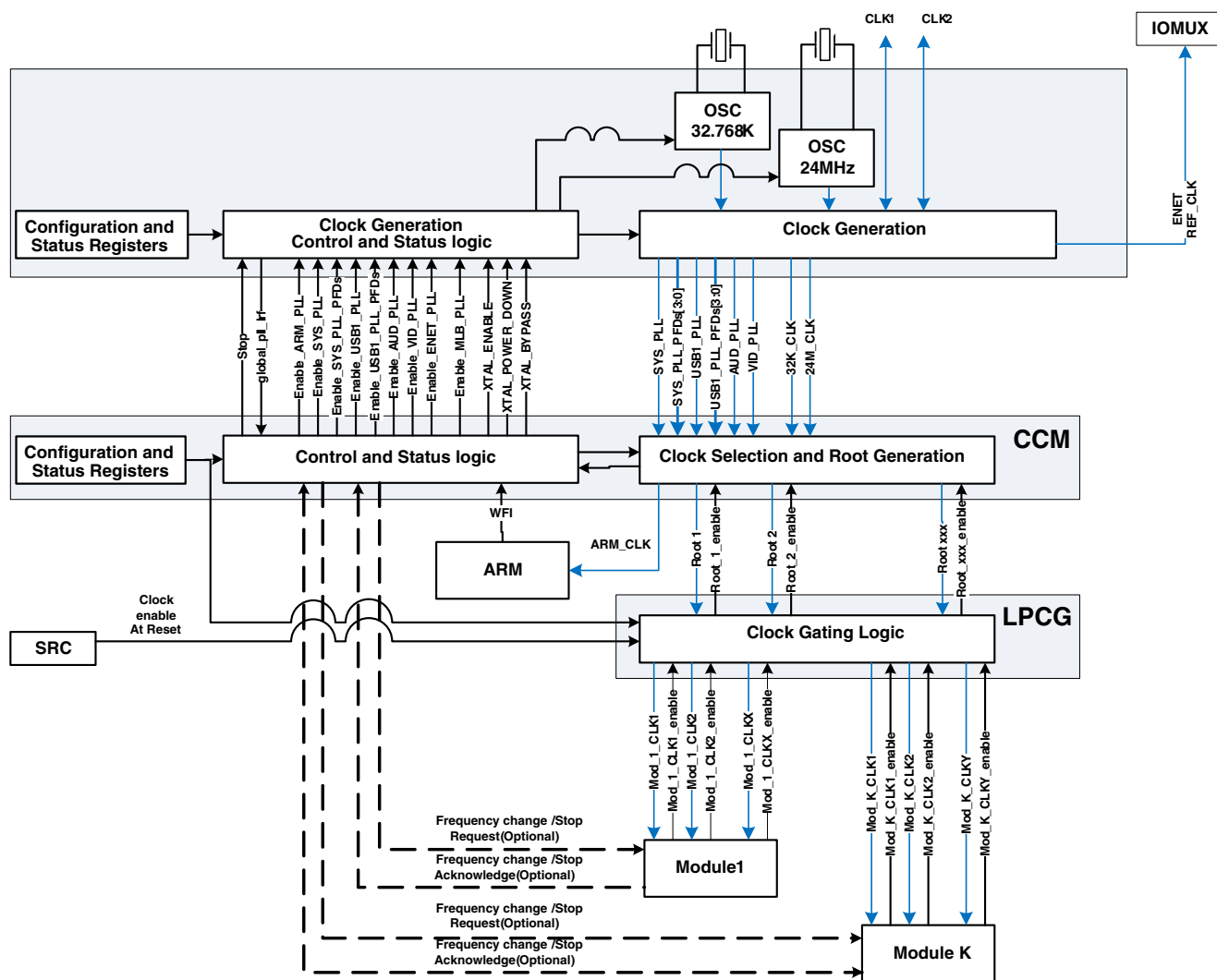
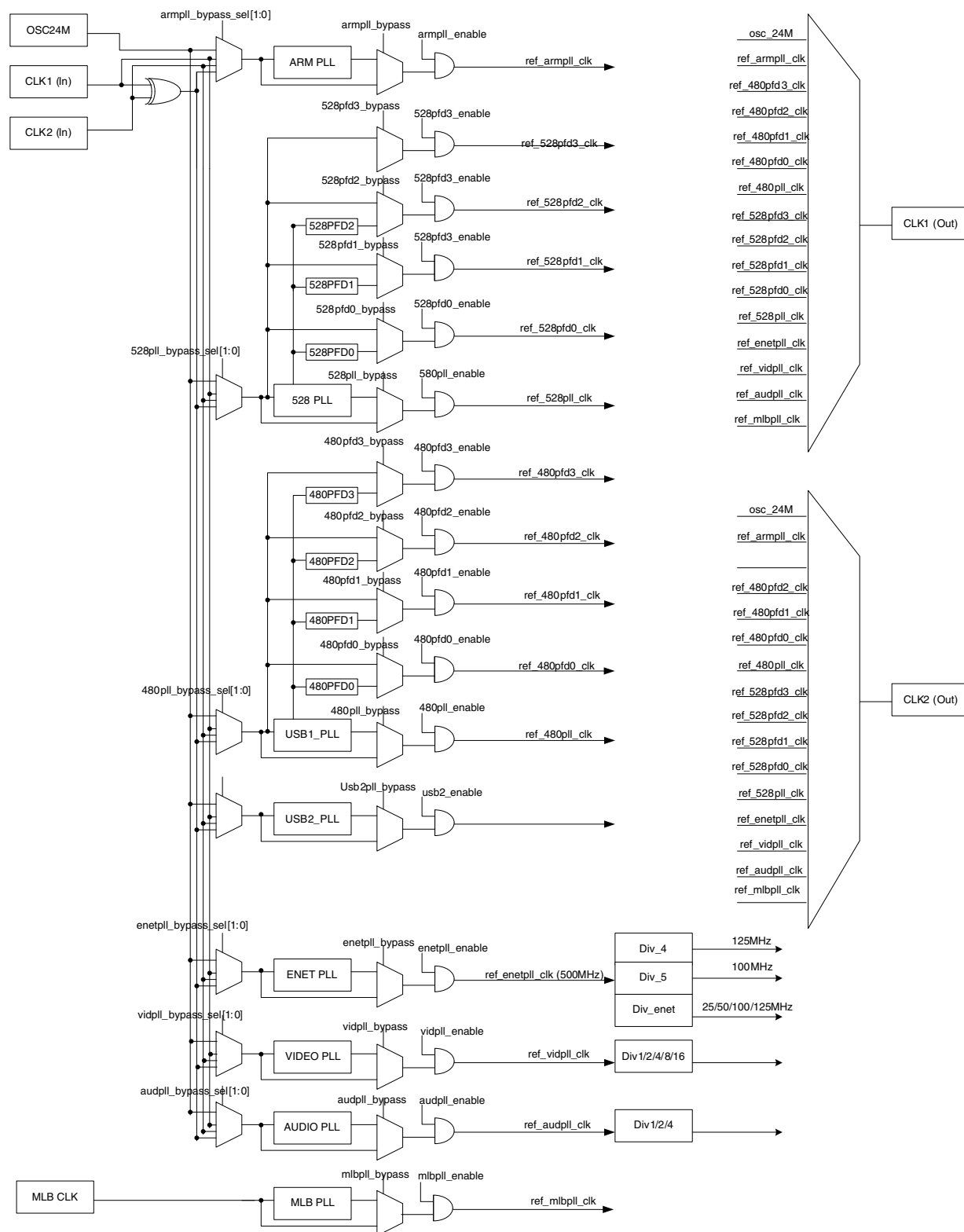


Figure 10-2. Clock Management System

A high level block diagram of the clock generation is shown in the figure below.



**Figure 10-3. Primary Clock Generation**

## 10.3.2 Clock generation

The clock generation section includes the components detailed in the following sections.

### 10.3.2.1 Crystal Oscillator (XTALOSC)

The Crystal Oscillator block is comprised of both the high frequency oscillator (typical frequency is 24 MHz) and the low frequency real time clock oscillator (typical frequency of 32.768 KHz). Each of these oscillators is implemented as a biased amplifier that, when combined with a suitable external quartz crystal and external load capacitors, implements an oscillator. Any of them may serve as an external clock input as well. See [Crystal Oscillator \(XTALOSC\)](#) for details of the XTALOSC block.

There is also a lower-power RC oscillator available on-chip that can be used as an alternative to the crystal oscillator. Because it has accuracy limitations it is intended to be used mainly in conjunction with the low-power modes. For more information please contact a Freescale FAE for pertinent application notes.

### 10.3.2.2 LVDS I/O ports

There are two LVDS I/O ports used for clock generation. These two low jitter differential I/O ports are provided to input and output clocks. They can take input clocks from outside of the SoC and provide them to the PLLs or to the other modules, or they can take the outputs of the PLLs and provide them outside of the SoC as a functional or reference clock.

### 10.3.2.3 PLLs

Seven PLLs are included in the clock generation section. Two of these PLLs are each equipped with four Phase Fractional Dividers (PFDs) in order to generate additional frequencies.

#### NOTE

Each PFD works independently by interpolating the VCO of the PLL to which it is connected. It effectively takes the PLL VCO frequency and produces  $18/N \times F_{vco}$  at its output where N ranges from 12 to 35. PFD is a completely digital design with no analog components or feedback loops. The frequency switch time is much faster than a PLL because keeping the base PLL

locked and changing the integer N only changes the logical combination of the interpolated outputs of the VCO. Note that the PFD not only enables faster frequency changes than a PLL, but also allows the configuration to be safely changed "on-the-fly" without going through the output clock disabling/enabling process.

The seven PLLs are listed below:

- PLL1 (also referred to as ARM\_PLL) - This is the PLL clocking the ARM core complex. It is a programmable integer frequency multiplier capable of output frequency of up to 1.3GHz. Note that this frequency is higher than the maximum chip supported frequency 1.0GHz.
- PLL2 (also referred to as System\_PLL or 528\_PLL) - PLL2 runs at a fixed multiplier of 22, producing 528MHz output frequency with 24MHz reference from XTALOSC. Besides the main output, this PLL drives three PFDs (PLL2\_PFD0...PLL2\_PFD2). The main PLL output and its PFD outputs are used as inputs for many clock roots. These do not require exact/constant frequency and can be changed as a part of dynamic frequency scaling procedure. Typically, this PLL or its PFDs are a clock source for internal system buses, internal processing logic, DDR interface, NAND/NOR interface modules, etc.
- PLL3 (also referred to as USB1\_PLL or 480\_PLL1) - PLL3 is used in conjunction with the first instance of USB PHY (USB0 PHY, also known as OTG PHY). This PLL drives four PFDs (PLL3\_PFD0...PLL3\_PFD3) and runs at a fixed multiplier of 20. This results in a VCO frequency of 480MHz with a 24MHz oscillator. The main PLL output and its PFD outputs are used as inputs for many clock roots that require constant frequency, such as UART and other serial interfaces, audio interfaces, etc.
- PLL4 (also referred to as an Audio PLL) - This is a fractional multiplier PLL used for generating a low jitter and high precision audio clock with standardized audio frequencies. The PLLs oscillator frequency range is from 650MHz to 1300MHz, and the frequency resolution is better than 1Hz. This clock is mainly used as a clock for serial audio interfaces and as a reference clock for external audio codecs. It is equipped with a divider on its output and can generate divided by 1, 2 or 4 from the PLL VCO frequency.
- PLL5 (also referred to as a Video PLL) - This is a fractional multiplier PLL used for generating a low jitter and high precision video clock with standardized video frequencies. The PLLs oscillator frequency range is from 650MHz to 1300MHz, and the frequency resolution is better than 1Hz. This clock is mainly used as a clock for display and video interfaces. It is equipped with dividers on its output and can generate clock divided by 1, 2, 4, 8 or 16 from the PLL VCO frequency.

- PLL6 (also referred to as ENET\_PLL) - This PLL implements a fixed  $20+(5/6)$  multiplier. With a 24MHz input, it has a VCO frequency of 500MHz. This PLL is used to generate:
  - 50 or 25 MHz for the external ethernet interface.
  - 125MHz for reduced gigabit ethernet interface.
  - 100MHz for general purposes.
- PLL7 (also referred to as USB2\_PLL or PLL2\_480) - This PLL provides clock exclusively to USB2 PHY (also known as HOST PHY). It runs at a fixed multiplier of 20, resulting in a VCO frequency of 480MHz with a 24MHz oscillator.

### 10.3.2.3.1 General PLL Control and Status Functions

PLLs configuration and control functions are accessible via individual per PLL and PFDs and global configuration and status registers.

Reference input clock for any of the PLLs except the MLB PLL could be selected individually by the BYPASS\_CLK\_SRC field of the PLL control register. See [CCM Analog Memory Map/Register Definition](#) for more information.

Each of the PLLs could be individually configured to "Bypass", "Output disabled" and "Power Down" modes.

When configured in "Bypass" PLL pass directly its input reference clocks to the PLL output. Bypassing the PLL is done by setting the BYPASS bit in the control register. For the PLL equipped with PFDs the input reference clock is also bypassed to all PFDs outputs.

When configured in output disabled mode (ENABLE=0), the PLL's output is completely gated and there is neither a bypass clock nor PLL generated clock that propagates to PLL output. Each PLL output has an individual "Output Enable" control bit. The PFDs are gated by the ENABLE bit of their associated PLL. Each PFD does have an associated clock gate bit that can be used to turn it off individually.

When configured in "Power Down mode" most of the PLL circuitry is switched off. Neither main PLL output nor PFD outputs are available in this mode.

When the related PLL is powered up from the power down state or made to go through a relock cycle due to PLL reprogramming, it is required that the related PFDx\_CLKGATE bit in CCM\_ANALOG\_PFD\_480n or CCM\_ANALOG\_PFD\_528n, be cycled on and off (1 to 0) after PLL lock. The PFDs can be in the clock gated state during PLL relock but must be un-clock gated only after lock is achieved. See the engineering bulletin, Configuration of Phase Fractional Dividers (EB790) at [www.freescale.com](http://www.freescale.com) for procedure details.

Individual PLL status is reflected in "PLL Lock" bits of the PLL control registers. PLL enable logic which monitors the register value change is implemented to gate off the PLL outputs during the "lock in" period.

Outputs are generated to be sent out by monitoring the individual PLL lock flags and filtering out any random initial edges.

Individual PLL Lock ready flags are first "ORED" with "enables" and then "ANDED" together to generate the global PLL lock ready flag that reflects status of all PLLs enabled in certain moment.

[CCM Memory Map/Register Definition](#) and [CCM Analog Memory Map/Register Definition](#) contains detailed descriptions of the memory mapped registers and control functions of the clock generation sub-module.

### 10.3.2.4 CCM

CCM includes:

- Clock root generation logic - This sub-block provides the registers that control most of the secondary clock source programming, including both the primary clock source selection and the clock dividers. The clock roots are each individual clocks to the core, system buses (AXI, AHB, IPG) and all other SoC peripherals, among those are serial clocks, baud clocks, and special functional clocks. Most of clock roots are specific per module.
- CCM, in coordination with GPC, PMU and SRC, manages the [Power modes](#), namely RUN, WAIT and STOP modes. The gating of the peripheral clocks is programmable in RUN and WAIT modes.

CCM manages the frequency scaling procedure for:

- ARM core clock - "on the fly" without clock interruption, by either shifting between PLL sources [PLL clock change](#) or by changing the divider ratio.
- Graceful changing of the DDR memory controller clock. See [MMDC handshake and Self refresh and Frequency change entry/exit](#) for more details.
- Peripheral root clock - by using programmable divider. The division factor can change on the fly without loss of clocks.

#### NOTE

Note: on-the-fly frequency changing for synchronous interfaces like serial audio interfaces (SSI, ESAI), Audio Sample Rate Converter (ASRC), or general purpose serial interfaces (UART, CAN) in general causes synchronization loss and should not be done.



### 10.3.2.5 Low Power Clock Gating unit (LPCG)

The LPCG block receives the root clocks from CCM and splits them to clock branches for each block. The clock branches are individually gated clocks.

The enables for those gates can come from four sources:

- Clock enable signal from CCM - This signal is generated depending on the power mode the system is in. For each power mode, it is defined in the software using the configuration of the CGR bits in CCM.
- Clock enable signal from the block - This signal is generated by the block based on its internal logic. Not every enable signal from the block is used. Each clock enable signal from the block can be overridden based on the programmable bit in CCM.
- Clock enable signal from the reset controller (SRC) - This signal will enable the clock during the reset procedure.

### 10.3.3 Peripheral components of clock management system

The figure found here shows the clock interface of a functional module in the system.

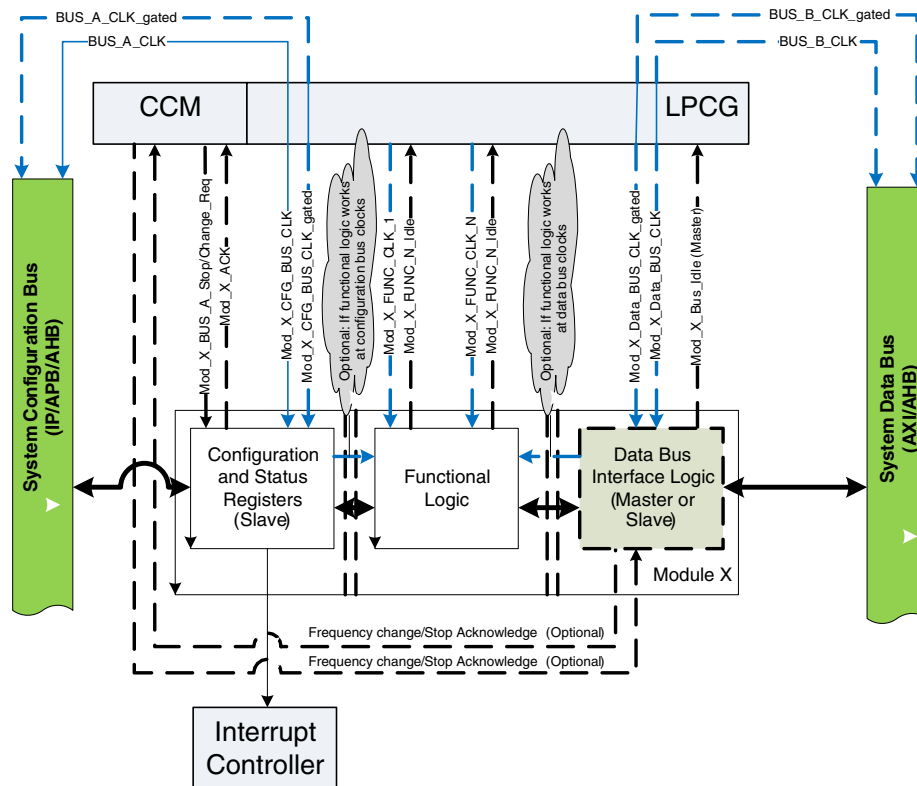


Figure 10-4. Clock interface of the functional module in system

### 10.3.3.1 Interface and functional clock

Each block within the SoC has specific clock input characteristic requirements. Based on the characteristics of the clocks delivered to modules, the clocks are divided into two categories: bus interface clocks and functional clocks.

The bus interface clocks have the following characteristics:

- They ensure proper communication between any block/subsystem and the system buses.
- In most cases, they supply the system interface and configuration registers of the block.
- A typical block has one system bus clock, but blocks with multiple interface clocks may also exist (that is, when a block is connected to multiple buses).
- The bus interface clocks are always fed by the outputs of the CCM/LPCG.
- Clock management for this type of clock is always implemented at the system level because it requires coordinated clock management between the block and system buses.

Functional clocks have the following characteristics:

- They supply the functional part of a block or a subsystem.
- Typically, these clocks are completely asynchronous and independent from the bus interface clock of the same block.
- A block can have one or more functional clocks. Some functional clocks are mandatory, while others are optional for its functioning. A block needs its mandatory clock(s) to be operational. The optional clocks are used for specific features and can be shut down without stopping the block activity.
- The functional clocks are fed either by a CCM/LPCG block functional clock output, or by some other clock source, such as a clock output of another block or an external signal coming from IOMUX.

### 10.3.3.2 Block level clock management

Each block in the system may also have specific clock requirements. Certain module clocks must be active when operating in some specific modes, or may be gated in some others. Generally, the activation and gating of the module clocks are managed by LPCG. Hence, the LPCG block must be programmed properly and, in case of hardware controllable clock gating, peripheral module should provide signals indicating when to activate and when to gate the module clocks.

The LPCG block differentiates the clock-management behavior for device modules based on whether the block can initiate transactions on the device interconnect (called master module), or if it cannot initiate transactions and only responds to the transactions initiated by the master (called slave module). Thus, two hardware-based clock-management protocols are used:

- Master protocol - Clock-management protocol between the CCM/LPCG and blocks that can be bus master
- Slave protocol - Clock-management protocol between the CCM/LPCG and slave modules

### 10.3.3.2.1 Master clock protocol

This protocol is used to indicate that a master module is ready to initiate a transaction on the device interconnect and requests specific (both functional and interface) clocks. The CCM/LPCG block ensures that the required clocks are active when the master module requests that the CCM/LPCG enable them. The module is said to be functional after the required clocks are activated.

Similarly, when the master module no longer requires the clocks, it informs the LPCG/CCM block and the LPCG/CCM can then gate the clocks to the module and all the clock precedents that are not used by other blocks. The master module is then said to be in clock-gated or partially clock gated mode.

Examples of modules supporting master clock protocol are GPU2D, VDOA and USDHC. Please see details in chapters describing these modules and in

### 10.3.3.2.2 Slave clock protocol

This hardware protocol allows CCM to control the state of a slave module. CCM informs the slave module, through assertion of a stop/change request, when its clocks (both interface and functional) can be changed or gated. The slave acknowledges the request and CCM is then allowed to gate or change the clocks to the block.

Similarly, a clock-gated slave module may need to be woken up because of some event or a service request from a master module. In this situation, CCM enables the clocks to the module and then de-asserts the stop request to signal the module to wake up.

Examples of modules supporting slave clock protocol are product="mx6dq mx6sdl">CAN, EPIT and GPT. Please see details in chapters describing these modules and in the CCM Module Enable Override Register (CCM\_CMEOR). See [CCM Memory Map/Register Definition](#) for more details.

The protocol in both "master" and "slave" cases is completely hardware-controlled, but software should configure the clock management behavior for the module in two places: in the CCM registers associated with the block and in the block configuration registers.

### **10.3.3.3 Clock Domain(s)**

A clock domain is a group of blocks fed by clock signals controlled by the same clock controls in CCM. By gating the clocks in a clock domain, the clocks to all the blocks belonging to that clock domain can be gated/activated, either by software control or by hardware control associated with block activity. Thus, a clock domain allows efficient control of the dynamic power consumption of the domain.

The device is partitioned into multiple clock domains and each clock domain is controlled by an associated group of clock gating cells within the LPCG block. This allows the CCM/LPCG to individually activate and gate each clock domain of the system.

Examples of clock domains are: Main AXI bus clock domain, GPU2D clock domain, etc.

### **10.3.3.4 Domain level clock management**

The domain clock manager can automatically (based on hardware conditions) and manage the bus interface clocks within the clock domain. The functional clocks within the clock domain are managed through software settings.

### **10.3.3.5 Domain dependencies**

A domain dependency is a hierarchical relationship between two clock domains. Clock domain "X" is said to depend on a clock domain "Y" when a block in clock domain "Y" provides services (or even just a clock) to a block in clock domain "X". As a result, clock domain "Y" must be active whenever clock domain "X" is active.

The dependency between two clock domains may also exist if one clock domain serves to ensure communication between two blocks (for example, the clock domain of the device interconnect).

## **10.4 Power management**

### 10.4.1 Centralized Components of Power Management System

The power generation and management system is built around the PMU and GPC blocks. A high level block diagram of the power management system in the SoC environment is shown in the figure below.

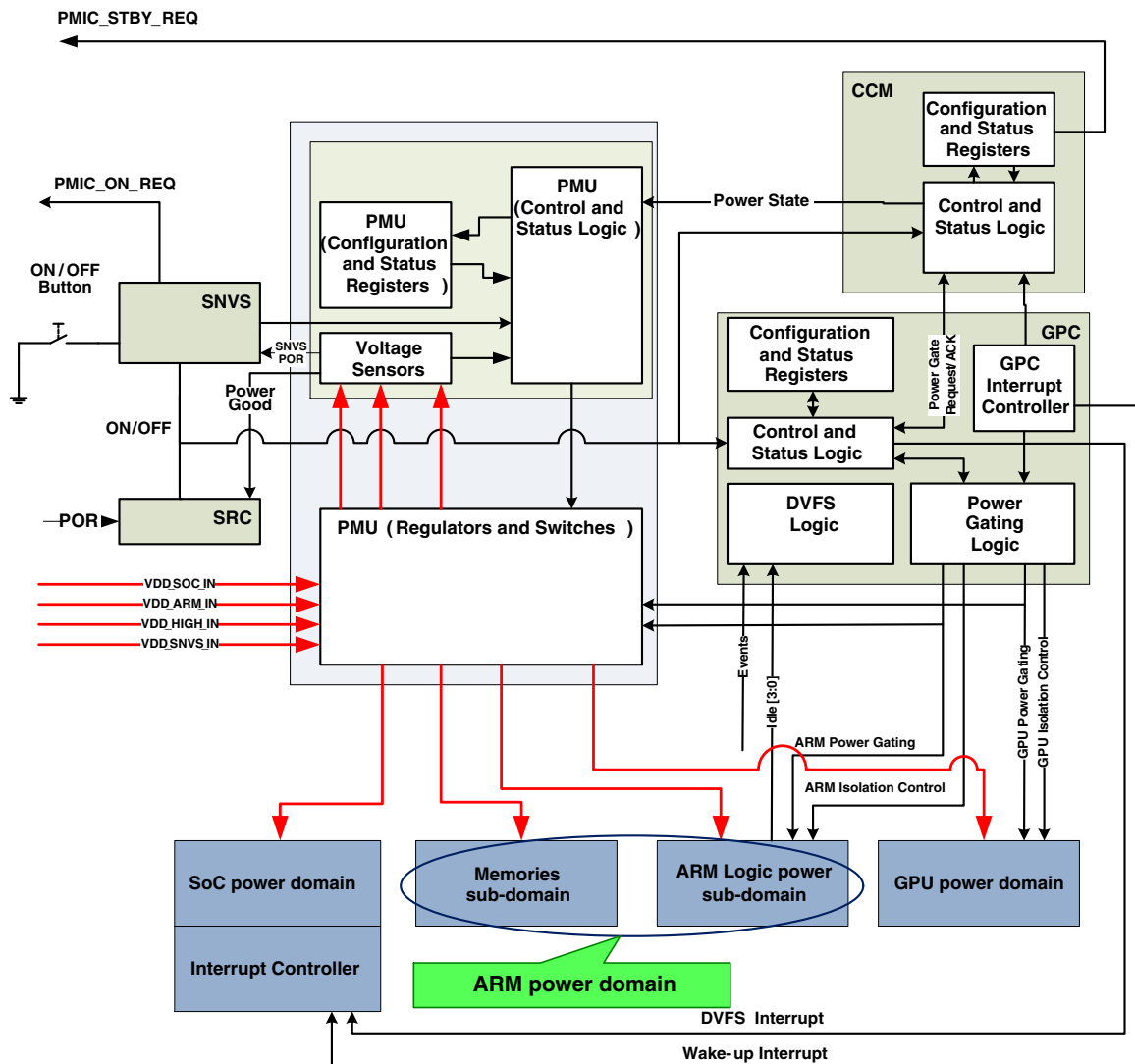


Figure 10-5. Power Management System

#### 10.4.1.1 Integrated PMU

The first component of the power management system, referred to as the integrated PMU, is designed to simplify the external power interface.

It consists of a set of secondary power supplies that enable SoC operations from just two or three primary supplies. The high level block diagram of the power tree, utilizing the integrated PMU, is shown below.

The integrated PMU includes the following components:

- Three Digital LDO regulators
- Two Analog LDO regulators
- USB LDO
- SNVS regulator
- Reverse well biasing

See [Power Management Unit \(PMU\)](#) for further details on integrated PMU functional description and programmability.

### 10.4.1.1.1 Digital LDO Regulators

The integrated PMU includes three digital LDO regulators: LDO\_ARM, LDO\_PU, and LDO\_SOC. These regulators provide power to the ARM\_Core power domain, the combined VPU and GPU power domain, and the rest of the SoC logic (except always-ON SNVS domain).

#### NOTE

The name "digital" only refers to the type of load. It is not related to the LDO design or feature set.

The digital LDO regulators can operate in the following modes:

- Internal Bypass - The regulation pass device (FET) is switched fully on, passing the external input voltage to the load unaltered. The analog part of the regulator is powered down in this state, removing any loss other than the IR drop through the power grid and the FET. Be aware that some time is required to switch from the internal digital bypass mode to the analog regulation mode. Typically it takes less than 100us. Please refer to [PMU](#) for further details on bypass and power gate configuration.
- External Bypass - The input and output of the regulator are shorted externally to the SoC. If operating in this configuration, enable the internal bypass early in the startup sequence before attempting high frequency/high power operation. Be aware that internal power gating is not available in this mode.
- Power Gate - The regulation FET is switched fully off, limiting the current draw from the supply. The analog part of the regulator is powered down, limiting the power consumption. The output voltage will fall to a level where the residual leakage

of the power FET balances with the leakage of the load. Power gating is applicable to ARM and PU power domains.

- Analog regulation mode - The regulation FET is controlled such that the output voltage of the regulator equals the programmed target voltage. The target voltage is fully programmable in 25mV steps.

These modes allow the regulators to implement voltage scaling and power gating, and allow bypass when an external high power efficient regulator is used as a direct source for some of the SoC loads.

These digital regulators also feature brownout detection, which is helpful to sense when supplies are starting to collapse. Note that the core will be interrupted on a brownout. Please see details in [Miscellaneous Control Register \(PMU\\_MISC2n\)](#).

For further details of LDO programming and configuration please refer to [Digital Regulator Core Register \(PMU\\_REG\\_CORE\)](#).

The power management system is built under assumption that in typical applications the single (and simple) shared power supply will be used for ARM core domain and SoC domain. The combined load gains some efficiency, especially in low power modes and saves BoM significantly.

The DVFS in a typical cost/complexity optimized application is considered by mean of internal LDO. In "full speed" modes LDO bypass is considered in both domains. The dynamic voltage scaling to low load workpoints for ARM domain is implemented by programming associated LDO.

In highly power-optimized systems, it is possible to use multiple external DCDC buck converters and bypass internal LDO for high power domains. The obvious trade-off is in the increased complexity of the external power supply components and the associated increase in the BoM and board design complexity.

#### 10.4.1.1.2 Analog LDO regulators

There are two analog LDO regulators used for general system purposes:

- LDO\_1P1 - The LDO\_1P1 linearly regulates down a higher supply voltage (2.8V-3.3V) to produce a nominal 1.1V output voltage. This regulator supplies digital portions of USB PHYs, PLLs, and the internal 24MHz oscillator.
- LDO\_2P5 - The LDO\_2P5 linearly regulates down a higher supply voltage (2.8V-3.3V) to produce a nominal 2.5V output voltage. The regular 2.5V LDO is combined with an alternate self-biased low-precision weak regulator which can be enabled for applications that need to keep the 2.5V output voltage alive during low power modes, where the main regulator and its associated global bandgap reference module are disabled to save power. The output of this weak-regulator is not

programmable and is a function of its input power supply as well as its load current. Typically with a 3V input power supply, the weak-regulator output is 2.525V and its output impedance is approximately 40Ohm. Special procedure is recommended to move load back and forth between the main and low power regulators. This regulator supplies most of the analog circuitry of the integrated PHYs, special I/Os (LVDS I/O, DDR I/O), and other analog and mix signal components integrated into the SoC.

#### 10.4.1.1.3 USB LDO

The USB\_LDO linearly regulates down the USB VBUS input voltages (typically 5V) to produce a nominal 3.0V output voltage. This regulator has a built in power-mux that allows the user to run the regulator from either one of the VBUS supplies when both are present. If only one of the VBUS voltages is present, the regulator automatically selects that supply. Current limit is also included to help the system keep the in-rush current within limits as required in USB 2.0 specification. This regulator supplies only low speed and full speed transceivers of USB PHYs.

#### 10.4.1.1.4 SNVS regulator

The SNVS regulator takes the SNVS\_IN supply and generates the SNVS\_CAP supply, which in turn powers the real time clock and low power section of the SNVS blocks. If VDDHIGH\_IN is present, then the SNVS\_IN supply is internally shorted to the VDDHIGH\_IN supply to allow coin cell recharging if necessary.

#### 10.4.1.1.5 Reverse well biasing

The reverse well biasing module on the SoC consists of a self-clocked/self-regulating charge-pump circuit, used to generate a negative bias voltage for the floating PWELL, and a low-power regulator. The low-power regulator is used to generate a positive bias voltage for the NWELL of the digital logic cells on the SOC power domain. Static leakage reduction can be achieved during SoC low-power modes through the use of these reverse well bias voltages. Please refer to the CCM\_CLPCR register in [CCM Memory Map/Register Definition](#) and PMU\_MISC0 in [PMU Memory Map/Register Definition](#) for details about well bias control and [Static](#) " for functional description of the reverse well biasing.

#### 10.4.1.2 GPC - General Power Controller

The GPC block provides hardware assistance to Dynamic Voltage Frequency Scaling (DVFS) and power gating, and includes the sub-blocks listed here.



- **DVFS load tracking block** - This block allows hardware tracking on the core load and generates an interrupt when a frequency change is requested. It does not generate any request for voltage and/or frequency changes made by a hardware signal. The frequency/voltage changing process requires interaction with the CCM block, as well as either the integrated PMU modules or the external programmable regulator. This process should be completed by either the CPU interrupt routine or a DMA transaction.
- **Power Gating Controller (PGC)** - This sub-block of GPC has the following functions:
  - Provides the user with the ability to switch off power to a target subsystem.
  - Generates power-up and power-down control sequences. This includes interaction with CCM/LPCG and SRC, and control for clock and reset generation for power domains affected by power gating.
  - Provides programmable registers that adjust the timing of the power control signals.
  - Controls the CPU power domain and the combined GPU/VPU power domain.
- **Wake-up interrupt controller** - This controller initiates the system wake-up from low power modes when only low frequency real time clock remains active, and thus the Generic Interrupt Controller (GIC) can not handle synchronous interrupt signals. Additional features are as follows:
  - Supports up to 128 interrupts
  - Provides an option to mask/unmask each interrupt
  - Detects interrupts and generates the wake up signal

See [General Power Controller \(GPC\)](#) for further details on GPC, its sub-blocks, and information on its functional description and programmability.

### 10.4.1.3 SRC - System reset Controller

The reset controller is responsible for the generation of all reset signals and boot configuration decoding.

It determines the source and the type of reset, such as POR, WARM, and COLD, and performs the necessary reset signal qualifications. SRC is capable of generating reset sequences in the following conditions:

- in interaction with external PMIC, based on external POR\_B signal and "power ready" signals generated by the integrated PMU
- or in interaction with the integrated PMU only, based on its "power ready" signal.

Based on the type of reset, the reset logic generates the reset sequence for either the entire SoC or for the blocks that are power-gated.

See [System Reset Controller \(SRC\)](#) for further details on SRC functional description and programmability.

### 10.4.1.4 Power domain(s)

A power domain is a group of blocks or sub-blocks fed by power sources controlled by the same power controls in GPC.

Some power domains can be split into a logic sub-domain and a memory sub-domain. The memory sub-domain in such case may contain two entities:

- Memory array(s) - Powered by a dedicated voltage rail enabling memory retention while core is OFF.
- Memory interface logic - Powered by the same voltage source as the logic sub-domain of the power domain.

Signals crossing power domain boundaries or sub-domain boundaries are passed through proper isolation and/or level-shifting cells to ensure robust operations of the SoC when some of domains are power gated or working at a reduced voltage.

Figure below shows the power domain interface in the system

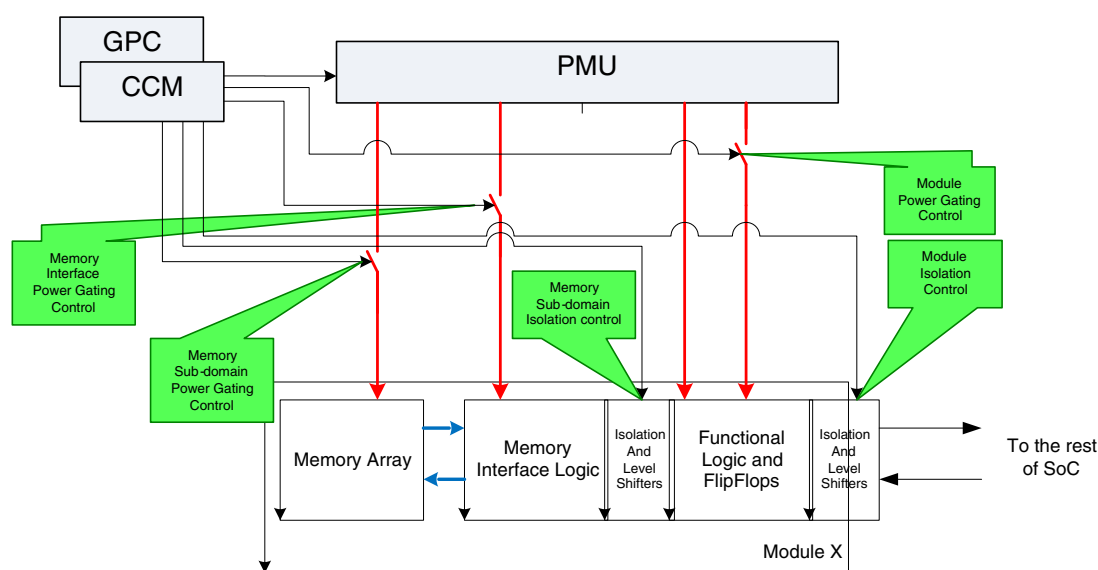


Figure 10-6. Power domain interface in system

#### 10.4.1.4.1 Power distribution

The power distribution tree is comprised of multiple power domains. The main power domains are:

- ARM - The ARM domain contains the ARM Core platform (except for memory arrays and interface logic). This domain can be supplied either from an integrated

power supply or from an external controllable regulator, preferably high efficiency DCDC converter.

- ARM Memory array - Memory arrays are connected to a separate and dedicated power domain (separated from the Main logic and ARM domain). In normal operation mode (functional, non-DVFS mode), the memory arrays domain voltage level should be kept equal to (same as) the rest of the core logic domains (Main, ARM). Please refer to the Datasheet for further information about voltage level difference between domains allowed in different power modes.
- PU domain - This domain contains the GPU2D and OpenVG engines.
- Display domain - This domain contains the display elements of the chip.
- SNVS/RTC low power domain - The SRTC domain contains only counter, comparator and compared data of the on-chip RTC. This domain should be supplied from an external single cell LiION battery and/or an external pre-regulated power supply.
- Analog domain - The analog domain contains the PLLs, LDOs and USB PHY. The domain supplies should be constant to allow continuous clock during any dynamic voltage scaling techniques. The digital supply should be provided from an internal regulator, and can be combined with the memory array supply. The analog supply should be provided from internal low noise regulator.
- Main SoC logic - The main SoC logic domain contains the rest of the logic of the SoC including most of the peripherals. It is supplied by an internal regulator.

From a DVFS and Power Gating standpoint, the following digital logic domains are affected:

- Cortex-A9 Core Platform - DVFS and power gating.
- ARM memories - Power gating only.
- PU - Power gating only.
- Display - Power gating only.

See table below for details of the i.MX 6Solo lite system power domains layout and dependencies.

#### **10.4.1.4.2 Domain Memory and domain logic state retention in case of Power Gating**

The following is the list of relevant memories and logic domains with the description of their state-retention support:

- Cortex-A9 Core Platform is sub-divided into three sub-domains listed below:
- Cortex -A9 Core Platform logic: The software state retention for all logic is implemented in this domain. That means that the content of relevant registers should be stored in some memory retaining its state (L2 cache for example) while the

logic domain is power-gated. Details on how to implement the software retention can be found in the Cortex-A9 Core Platform TRM.

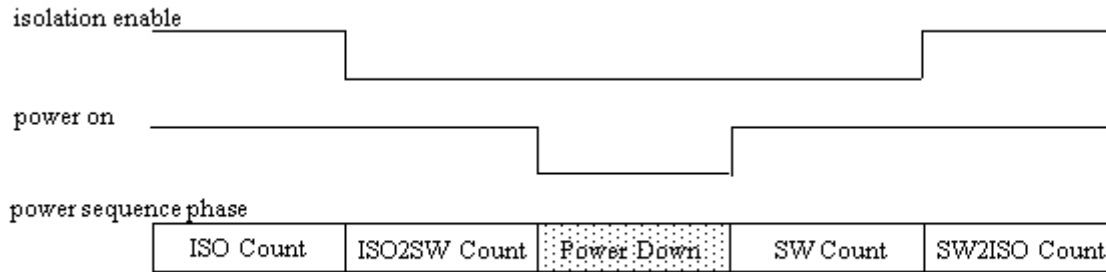
- Cortex-A9 Core L1 memories - No retention. The L1 memories have a dedicated supply on the package (VDD\_CACHE\_CAP) which should be connected to the Cortex-A9 Core Platform supply. The L1 cache should be flushed prior to power gating in order to allow powering up of the CPU at the same state as before power gating.
- Cortex-A9 Core L2 memories - hardware state-retention since its supplies are driven by the SoC supplies.
- PU (GPU2D and OpenVG): These modules can be power gated (together) independently of Cortex-A9 power gating, because it resides on a separate power domain. State retention is not supported for PU logic nor for their internal memories.
- SoC - hardware state-retention in Standby mode. Reverse Well Biasing is applicable for this domain.
- SNVS\_LP - hardware state-retention even when SoC supplies are removed.

#### 10.4.1.4.3 Power Gating Domain Management

The following bullets provide the sequence required for power-gating the relevant power-domains:

##### 10.4.1.4.3.1 Cortex-A9 Core Platform

1. Copy through software all the Core configuration registers to a powered-on memory
2. Configure the GPC/PGC CPU registers in [PGC Memory Map/Register Definition](#) as follows to power-down the core on the next "WFI" instruction:
  - Configure the GPC/PGC PGC\_CPU\_PDNSCR Register ISO and ISO2SW bits. These bits determine the delay between the power-down request to enabling the platform isolation and the platform isolation to the actual power-off switch to the supplies accordingly.
  - Configure the GPC/PGC PGC\_CPU\_PUPSCR Register SW and SW2ISO bits. These bits determine the delay between the power-up request to the actual power-up of the supplies and the last to the platform isolation disabling.
  - Configure the GPC PGC PGC\_CPU\_CTRL PCR bit to allow the power down of the platform
  - Cortex-A9 Core Platform should execute a "WFI" instruction.



**Figure 10-7. Cortex-A9 Core Platform isolation and power on switch flow**

#### 10.4.1.4.3.2 GPU2D

1. Configure the CCM CGR bits ([CCM Memory Map/Register Definition](#)) to disable the GPU2D clocks.
2. Configure the GPC/PGC GPU registers ([GPC Memory Map/Register Definition](#)) as follows to power-down isolate the GPU2D logic from the rest of the SoC logic:

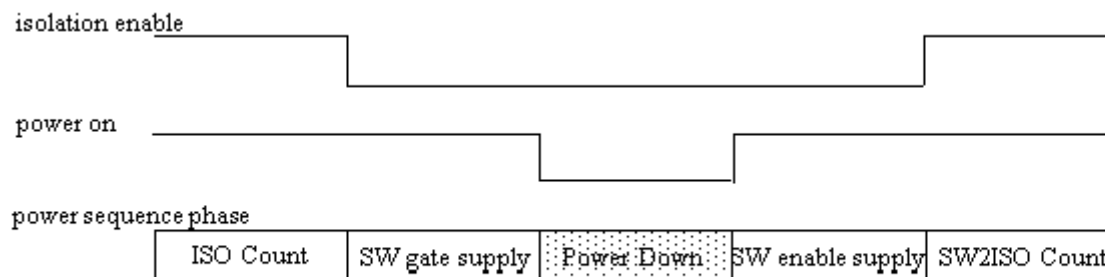
Configure the GPC/PGC PGC\_GPU\_PDNSCR Register ISO bits. These bits determine the delay between the power-down request to enabling the PU isolation.

Configure the GPC/PGC PGC\_GPU\_PUPSCR Register SW2ISO bits. These bits determine the power-up request to the PU isolation disabling.

Configure the GPC PGC PGC\_GPU\_CTRL PCR bit to allow the power down of the platform.

1. Software should wait PDNSCR x IPG\_CLK cycles before next step which is powering down the PU
2. Configure the PMU PMU\_REG\_CORE ([PMU Memory Map/Register Definition](#)) to gate off the PU regulator. The exit sequence should be in the reversed order i.e. restore power first and enable the clocks last.

When powering up the PU domain a delay may need to be added between enabling the PU LDO and disabling the PU isolation.



**Figure 10-8. GPU and Display isolation and power on switch flow**

## 10.4.1.4.3 SoC

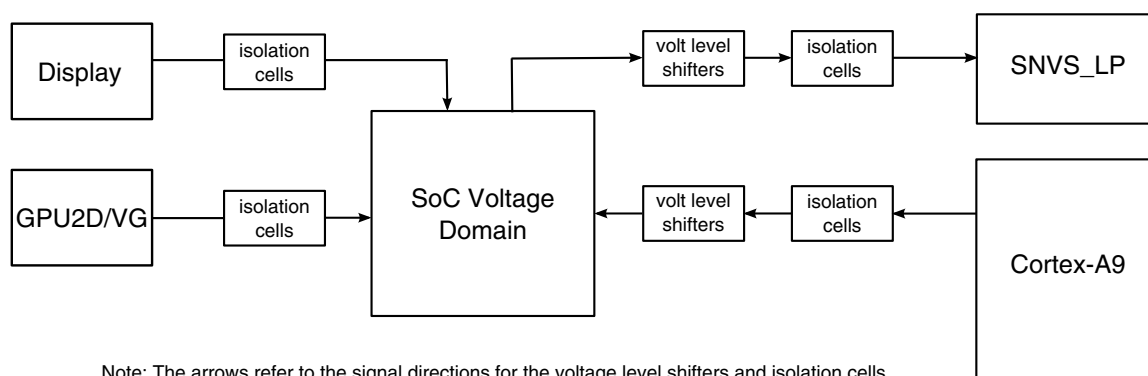
For additional power reduction it is possible to do the following:

- Power-down the internal oscillator by configuring the following bits CCM\_CCR[COSC\_EN] ([CCM Memory Map/Register Definition](#)). This can be done only in case there is no dependency on 24MHz XTAL for wake-up.
- Enable reverse well biasing by configuring the CCM\_CLPCR[WB\_PER\_AT\_LPM] bit ([CCM Memory Map/Register Definition](#)).
- It is possible to turn off and turn on the PMIC supplies to the SoC even when the SoC supplies are off. Since SNVS\_LP is powered through an "always on" supply, configuring the SNVS\_LP DP\_EN to "1" allows changing the PMIC\_ON\_REQ pad (SoC on/off supply indication to the PMIC) through the RESET\_IN\_B pad.

## 10.4.1.4.4 Power Gating domain dependencies

There are 3 power domains that need to be isolated in different power-down cases:

- Cortex-A9 Core Platform - Isolation needs to be enabled before power-down. This is taken care of automatically once CCM and PGC are configured and the Cortex-A9 Core Platform executes the "WFI" instruction.
- GPU2D and Display - Isolation needs to be enabled before power-down. This should be taken care through software configuration of the PGC.
- SNVS\_LP - Different from the 2 cases above the SNVS\_LP isolation isolates the signals coming from the SoC to the SNVS\_LP. This is required for saving the contents of the SNVS\_LP. (such as the real-time clock) The isolation is activated in 2 ways:
  - Automatically through the power-fail detector in the PMU
  - Through software configuration



**Figure 10-9. Isolation cells and Voltage level shifters placing**

### 10.4.1.5 Voltage domains

The list found here states the different voltage domains and their scalability in regarding to power-saving in dynamic and static scenarios.

- ARM - Cortex-A9 Core Platform including L1 cache - Scalable voltage in both dynamic and static scenarios
- SoC and PU - Scalable voltage only in static scenarios
- ANALOG components including LVDS and PLLs - Fixed voltage
- I/O - Fixed voltage
- SNVS\_LP - Fixed voltage

### 10.4.1.6 Voltage domain management

#### 10.4.1.6.1 Dynamic

##### 10.4.1.6.1.1 DVFS

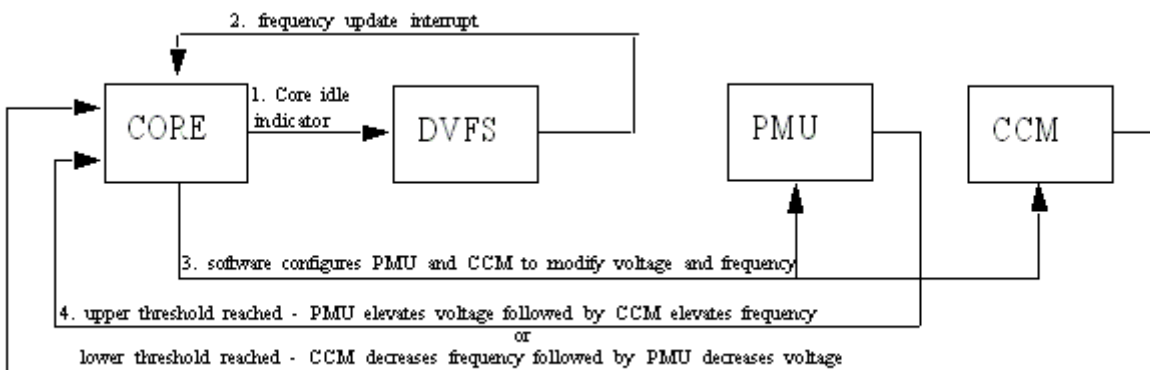
Dynamic Voltage and Frequency Scaling (DVFS) is a well-known technique to reduce power consumption in mobile devices. In order to improve power saving efficiency, DVFS is applied on the ARM core voltage domain.

In this scenario the Dynamic Voltage and Frequency Scaling (DVFS) block is used. More details can be found in [DVFS-CORE \(DVFS\)](#). The DVFS block can monitor separately the IDLE indication of each of the Cortex-A9 Cores. The DVFS performs the following:

- Simple, non-overlapping averaging providing a level-based average index of the tracked CPU load
- Sums the CPU load and the load detected from additional load indicators weighted according to software configurations
- Calculates an exponential moving average of the tracked load
- Provides up, down and "panic" threshold comparators and counters for generating interrupts to the Cortex-A9 Core Platform
- Frequency pattern generator is able to manage the frequency update requests periodically

The DVFS interrupts are then forwarded to the Cortex-A9 Core Platform. The Cortex-A9 Core Platform reacts according to the DVFS interrupt types. The following are 2 examples for handling different DVFS interrupt types:

- Upper threshold is reached, meaning that the Core is heavily loaded and the core frequency needs to be increased according to the following flow:
- Configure PMU to raise the core voltage
- Wait until the voltage is stable
- Configure CCM to raise the frequency.
- Lower threshold is reached, meaning that the Core is "IDLE" most of the time and the core frequency can be decreased according to the following flow:
- Configure CCM to reduce the frequency.
- Configure PMU to lower the core voltage.



**Figure 10-10. High Level DVFS working flow**

#### NOTE

PMU can not be used to change the voltage in case the LDO is bypassed.



#### 10.4.1.6.1.2 Voltage Scaling

A simplistic way to reduce power consumption in dynamic scenarios is to scale down the ARM, SoC and PU voltage according to the allowed voltage points and corresponding frequencies specified in datasheet.

#### 10.4.1.6.2 Static

##### 10.4.1.6.2.1 Standby Leakage reduction (SLR)

Standby leakage reduction is a power-management technique utilizing:

- Reduced supply voltage for relevant domains
- Reverse well-biasing in STOP, WAIT, or Deep Sleep Mode (DSM) modes

With SLR, the device switches into low-power active system modes automatically or in response to user requests during system Stop, Wait, or DSM modes (that is, in situations when no application is started and no system activity is presented).

When applying SLR, the system remains in the lowest static power mode while retaining logic and memory states. This technique trades static power consumption for wake-up latency while maintaining fast system response time suitable for most applications.

See CCM Control Register (CCM\_CCR), CCM Low Power Control Register (CCM\_CLPCR) and PMU Miscellaneous Register 0 (PMU\_MISC0) for further details on SLR programmability options.

The following describes the flow for applying standby voltage and reverse well bias to the SoC:

- Configure the external PMIC standby voltage, refer to chip datasheet.
- Configure CCM\_CCR[RBC\_EN] bits to bypass and disable PMU regulators in the next ARM "WFI" execution.
- Configure CCM\_CCR[REG\_BYP\_COUNT] bits to allow proper voltage restoration by the external PMIC when exiting standby.
- Cortex-A9 Core Platform executes the "WFI" instruction that completes the software sequence putting the SoC into low power mode
- After that, the reverse well bias will be applied automatically (if enabled) with appropriate delay. Please refer to CCM\_CLPCR and CCM\_CCR Registers in CCM for further information on reverse bias enabling and counters configuration.

#### 10.4.1.6.3 Voltage domain dependencies

When applying voltage changes for power saving the following voltage domain limitations should be taken into account:

- In dynamic scenarios the SoC and PU voltage supplies should not differ
- In dynamic scenario's the ARM supply should not be higher than the SoC supply by more the 50mV.

The above limitations are also mentioned in datasheet.

#### 10.4.1.6.4 IO voltage

#### 10.4.1.7 system domains layout

The following table describes the different aimed power modes for :

#### NOTE

Wakeup time is the hardware perspective, and doesn't reflect the time it takes software to resume drivers and perform system operations.

**Table 10-1. Aimed power modes**

Mode	Description (Status of main power domains)	Wake-up capability	Wakeup time	Applicable use case
RUN	Power supplies are on, all clocks are on.	N/A	N/A	All
WAIT	Power supplies are on. ARM executes WFI command, Option to gate off clocks for specific modules based on programmable bits.	Based on interrupt from any module	Immediate (ARM exit WFI)	IDLE process of operating system.
STOP	Power supplies are on. ARM executes WFI command. GPU is in power gating PLLs are off, hence all system clocks are off.	Based on interrupt from modules that can track operation with low frequency clock- like Keypad press, GPIO, Timer, etc.	~50us	Suspend state of operating system - system is on but is waiting for operation.

*Table continues on the next page...*

**Table 10-1. Aimed power modes (continued)**

Mode	Description (Status of main power domains)	Wake-up capability	Wakeup time	Applicable use case
Standby	<ul style="list-style-type: none"> <li>- PLLs disabled</li> <li>- Low voltage could be applied to ARM and SoC power inputs</li> <li>- Chip regulator bypass if reduce supply voltage,</li> <li>- Well Bias could be activated</li> <li>- CPU regulator disabled for power gating</li> <li>- PU regulator is disabled in software</li> <li>- xtal enabled</li> </ul>	Same as above	~220 us	Low power standby mode with limited reaction on external events
Deep Sleep	Same as above but with OSC24M disabled	Interrupts from modules those could generate asynchronous interrupts (Keypad, GPIO, SNVS)	~1400us	
SRTC	Only SNVS domain and OSC32K keep alive	<p>SRTC security alert / SRTC timer expire.</p> <p>Based on each one of those, SRTC module will generate request to PMIC to perform POR sequence and walk-up the system. The system will go through boot process.</p> <p>PMIC can also receive press on "ON" button to walk-up the system. This should be connected directly to PMIC.</p>	<p>~1700us to WARM boot</p> <p>COLD boot, depends on boot procedure and boot device</p>	

There is a single hardware signal coming into PMU which sets the PMU in either of two "STOP" states. The STOP state is implemented is controlled by the PMU\_MISC0[STOP\_MODE\_CONFIG] bit (See [PMU Memory Map/Register Definition](#)). It is recommended that the blocks be configured for safe powerdown/up through the registers before asserting the stop\_mode signal. Blocks not described in the section below are unaffected by stop\_mode.

If the stop\_mode\_config is set to zero, thus in the STOP mode all blocks powered down in minimum power configuration.

If the stop\_mode\_config is set to one, thus in the STOP mode some of the blocks remain powered and in different states as defined in the table below.

**Table 10-2. STOP mode configuration**

Block	STOP_MODE_CONFIG=0	STOP_MODE_CONFIG=1
reg1p1	off	on
reg2p5	off	on
reg3p0	off/on depending on vbus. Uses crude local reference if vbus is present	off/on depending on vbus . Uses analog central bandgap if VBUS is present.
reg_core	bypassed if not power gated.	bypassed if not power gated
reg_pu	bypassed if not power gated.	bypassed if not power gated.
reg_soc	bypassed	bypassed
bandgap	off	functional
temp_sensor	off	off
well_bias	hardware controlled	hardware controlled
All PLs	off	off
OSC24M	off	Controlled by CCM configuration
LVDS clock in/out CLK1 and CLK2	off	off

## 10.4.2 Power management techniques

The device supports the power-management techniques with the features found here.

- Partitioning of the device into voltage, power, clock, and reset domains
- Domain isolation that allows flexible configurations of domains on/off states to form use cases targeting various applications
- Clock tree with selective clock-gating conditions and almost independent clock roots
- Power, reset, and clock control hardware mechanism to manage sleep and wake-up dependencies of power domains
- Software-controllable and hardware-controllable clock gating for functional modules and buses
- Memory retention and state retention capability (Software State Retention for ARM A9) for preserving memory contents and device state in low-power modes
- Dynamic Voltage and Frequency Scaling (DVFS) support for the ARM A9 processor cluster
- Support for low-power device modes input/output (I/O) pad configuration for minimum power
- Variety of operating modes to optimize device performance and wake-up times
- Thermal monitoring and thermal aware performance management

Many of the low power features are fully or partially software controllable and can be configured for the specific requirements of a target system.

Combining these techniques, the system designer may meet tight requirements of low-power standby and operational modes while maintaining high performance for time-critical tasks.

### 10.4.2.1 Power saving techniques

The table below lists power saving techniques supported by the SoC in their connection to different components of power consumption.

**Table 10-3. Power saving design/architecture and power saving techniques**

Techniques	Active SoC Power	Standby SoC Power	System Power
Temperature Monitoring, and active frequency throttling	√		
Cortex-A9 Core Platform DVFS	√		
Cortex-A9 Core Platform SRPG (Software)		√	
Cortex-A9 Core Platform Power Gating		√	
GPU2D Power Gating	√	√	
Clock gating (automatic dynamic and forced)	√		
Integrated PMU (IR drop, efficiency, accuracy)	√		√
C4 package (IR drop, thermal)	√		
Display Backlight optimization (SW)			√
Architecture: L2 cache, Video / Audio / Graphics acceleration	√		
Architecture: USB, LVDS integration			√
Low Power DDR: LPDDR2, LV-DDR3			√

### 10.4.2.2 Thermal-aware power management

The temperature sensor block (TEMPMON) implements a temperature sensor/conversion function. The block features an alarm function that can raise an interrupt signal if the temperature is above a specified threshold.

Software may implement temperature aware DVFS for the ARM domain and the GPU domain, as well as temperature aware frequency scaling for other system components to ensure that both the frequency and voltage is lowered when the die temperature is above the specified limit.

Software may also implement temperature aware task scheduling to ensure that non-critical tasks are suspended when the die temperature is above the specified limit.

See [Temperature Monitor \(TEMPMON\)](#) for further details on temperature monitor functions and programmability options.

### 10.4.2.3 Peripheral Power management

#### 10.4.2.3.1 Main memory power management

Main system memory, DDR3, and LPDDR2 are some of the most power-hungry system components, but the SoC provides several options to manage DDR power.

Automated power saving modes are supported by the MMDC hardware. This feature allows the DDR memory to automatically enter self-refresh mode when there are no DDR accesses for a configurable time. The default setting is 1024 clock cycles which can be optimized based on the customer use case and application.

See [Power Saving and Clock Frequency Change modes](#) for further details on MMDC power saving features and programmability options.

Software may support DDR frequency scaling. Automated frequency changing procedure is supported by MMDC and CCM modules.

#### NOTE

DDR frequency changes cost extra time and power. Slowing requestors while keeping DDR at full speed may increase total system power. Software may also implement Cooperative Dynamic Frequency Scaling in order to keep the system balanced, (that is, keep the system in balance when DDR throughput is equal or slightly higher than total amount of requests generated by all requestors).

Reducing the DDR frequency while in DLL-ON mode may be not efficient because:

- Reduction in DDR frequency will cause bus duty cycle to increase and thus reduces chance of automatic MMDC power saving (place memory into SR).
- Total amount of read/write operation does not change (power is per-operation).
- The termination is active longer, though, lowering frequency from 528Mhz to 400MHz or below may enable lowering drive strengths and termination.

When possible at lower performance use cases, software may switch DDR3 to DLL-off mode. This allows it to greatly reduce DDR3 frequency and thus disable or reduce termination and drive strength, which significantly reduces the power consumption of the DDR3 interface.

A good strategy for many types of workloads is to combine most activity in bursts (natively possible, for example, for typical multimedia applications, communication, etc.) and run this segment at maximal speed, then switch to DLL-OFF mode to support background activity (communication, display refresh, housekeeping).

The DRAM Interface power dissipation depends on many variables, however, proper termination and drive strength is key for power and thermal performance. Memory and controllers provide a host of programmable options for the drive strength of the output buffers and for the on-die termination impedance.

The ideal settings for drive strength and ODT also depend on the clock frequency to ensure that inter-symbol interference (ISI) effects are not introduced.

DDR PHY power is proportional to the amount and type of bus activity.

In cases where the DDR is placed into self-refresh, software can configure DDR I/O to be floated or lowered to the minimum drive allowed by JEDEC.

Modifying the DDR drive strength must be done by code that is executing from a memory region other than DDR (for example, IRAM). No access to DDR (including page table walks, cache misses, alternate bus master accesses, etc.) is allowed while the DDR I/O pads are being re-configured.

#### **10.4.2.3.2 Video-Graphics system power management**

#### **10.4.2.3.3 IO power reduction**

SW should configure IO to low power modes:

- PHYs - make sure that all unused PHYs are placed to lowest power state. Please refer relevant chapter for further information about different PHYs
- Digital IOs - Make sure all unnecessary PU/PD are disabled and IO are switched to either minimal drive strength or to input mode (when applicable)
- Set DDR type IO to CMOS mode if possible, specifically RGMII segment
- Disable all unused LVDS pads (ANALOG, LDB)

### **10.4.3 Examples of External Power Supply Interfacing in the i.MX 6SoloLite based systems**

This section presents the examples of external power supply interfacing to the chip.

The scenario based on integrated PMU system is presented below. This scenario minimizes BoM and board design complexity.

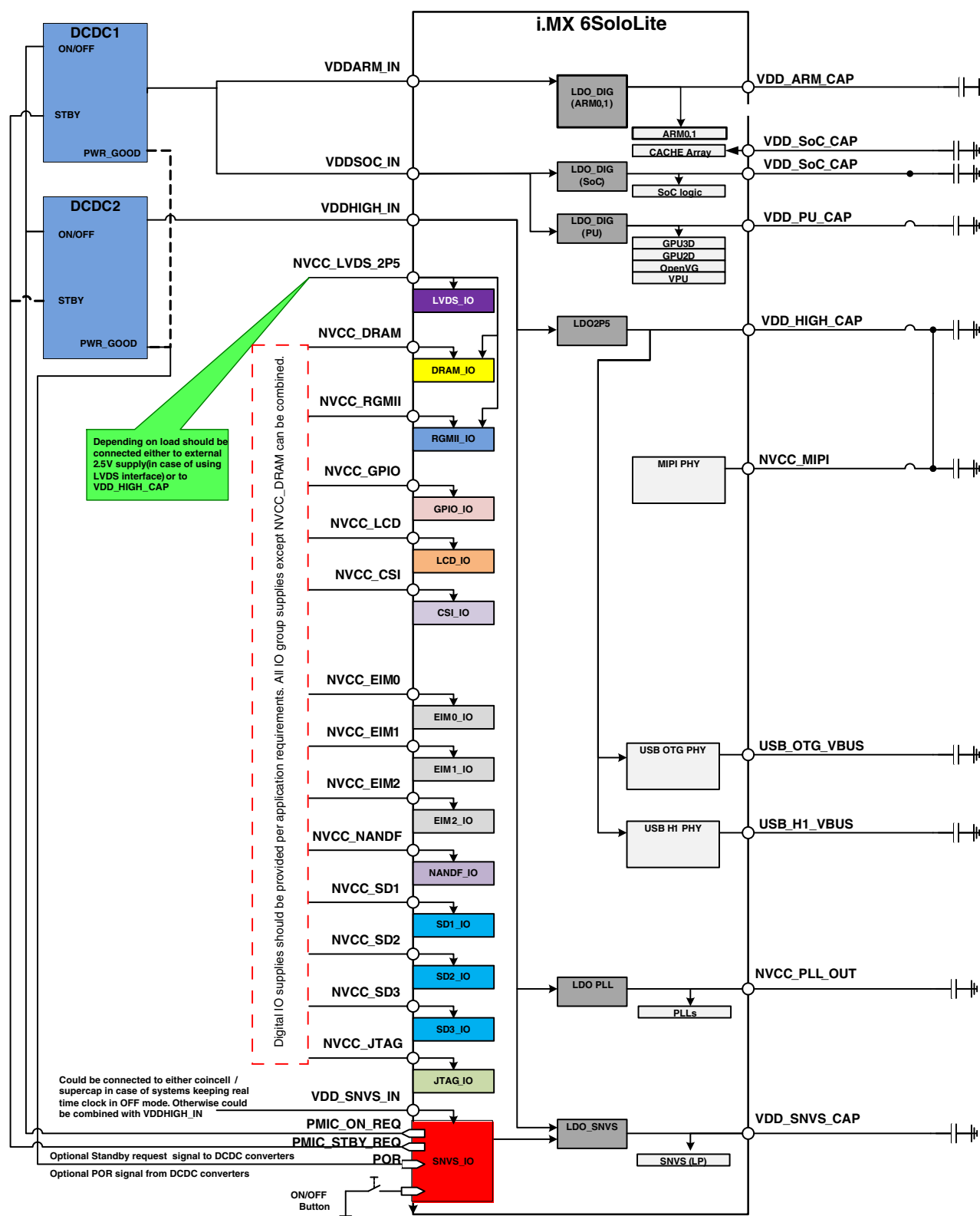


Figure 10-11. Supplying i.MX 6SoloLite power using integrated PMU



The scenario based on external programmable regulators is presented below: This scenario could be used in highly power optimized systems.

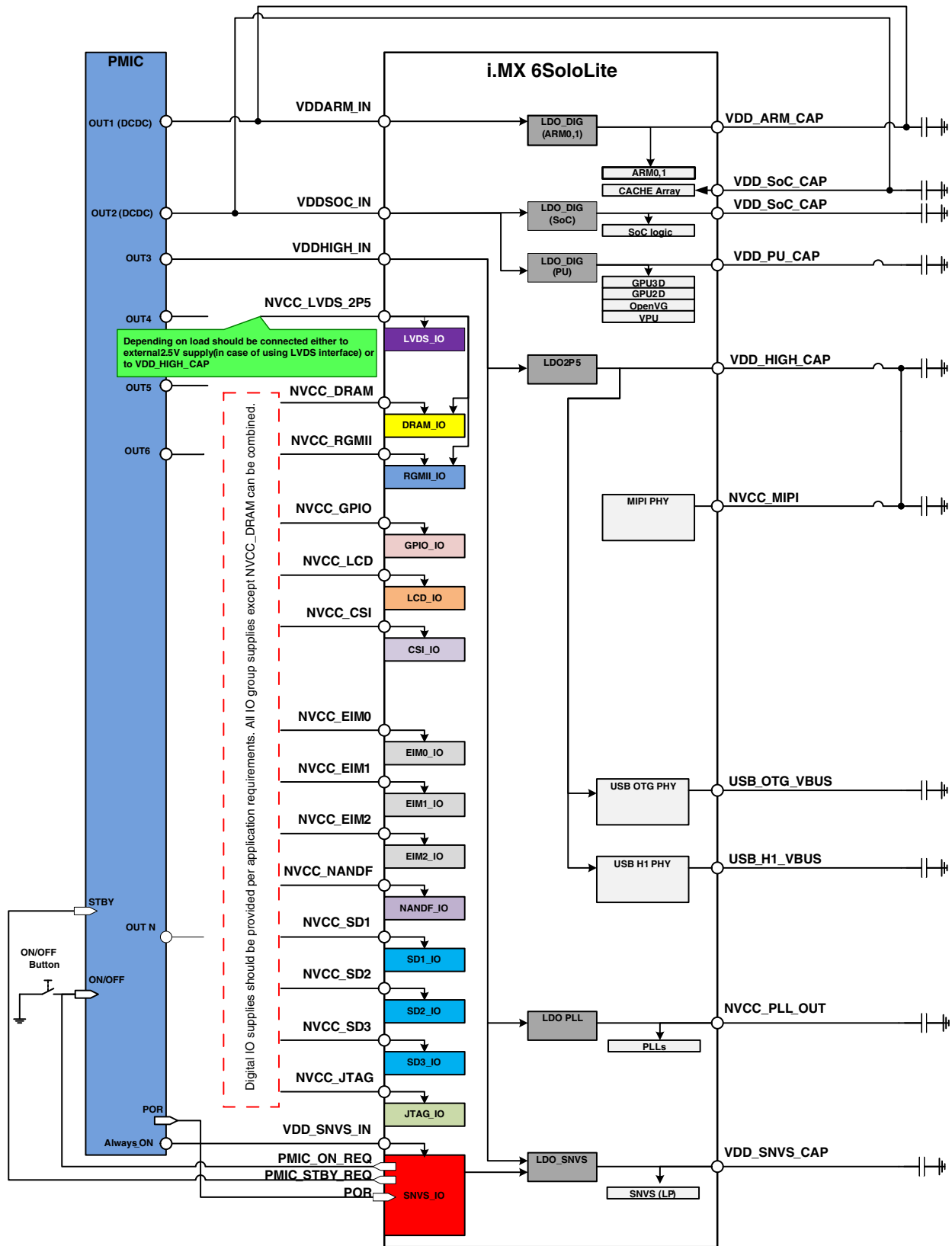


Figure 10-12. Supplying i.MX 6SoloLite Power Using External Programmable PMIC

## 10.5 ONOFF (Button)

The chip supports the use of a button input signal to request main SoC power state changes (i.e. On or Off) from the PMU.

The button is used to power On and Off the SoC using an always-on (e.g., coincell battery-backed) power domain.

- When the chip main power supply is Off, a button press greater in duration than 750 ms asserts an output signal to request power from a power IC to power up the SoC.
- When the chip main power supply is On, a button press between 750 ms and 5 seconds will send an interrupt to the core to request that software bring down the SoC safely. Software may respond to the interrupt by saving the processor state and then setting a control bit that requests to the power IC the removal of the main power supply.
- Button presses greater than 5 seconds, when the SoC is powered, results in a direct hardware power down request signal to the power IC without providing a software interrupt first. This long button press initiates a hardware-enforced mode which is applicable when software is unable to power Off the device.

The button is connected to input RESET\_IN\_B and the power IC is connected the chip output PMIC\_ON\_REQ. The chip must have TEST\_MODE deasserted. An always-On supply (e.g. coincell) is needed in the system for this feature.



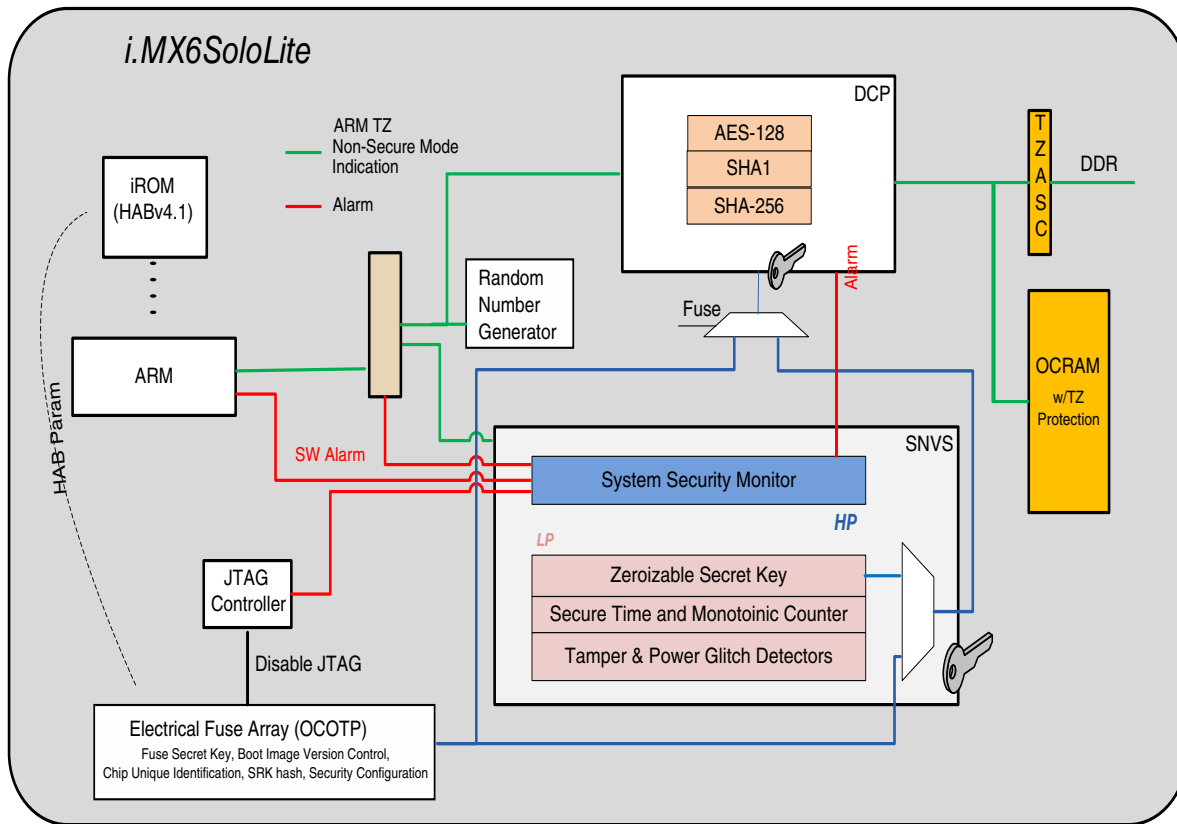
# Chapter 11

## System Security

### 11.1 Overview

Security is a common requirement for platforms built using the i.MX 6SoloLite, although the specific needs vary greatly depending on the platform and market. The type and cost of assets to be protected on a portable consumer device are very different from those to be protected on automotive or industrial platforms, and the same applies to the kind of attacks and level of resources threatening those assets. The platform designer must select an appropriate set of counter measures to meet the relevant platform security needs.

The following figure shows a simplified diagram of the security subsystem.



**Figure 11-1. Security subsystem (simplified)**

For the platform designer to meet the requirements for each market, the i.MX 6SoloLite incorporates a range of security features which can be used individually or in concert to underpin the platform security architecture. Most of the i.MX 6SoloLite security features provide protection against particular kinds of attack and can be configured at various levels according to the required degree of protection. These features are designed to work together and can be integrated with appropriate software to create defensive layers. In addition to protection features, the i.MX 6SoloLite includes a general purpose accelerator to enhance the performance of selected industry standard cryptographic algorithms.

The following is an introduction to the i.MX 6SoloLite security components.

- TrustZone (TZ) Architecture in the ARM Cortex A9 Platform, TrustZone aware Interrupt Controller (GIC) and TrustZone Watchdog Timer (WDOG-2)
- TrustZone Address Space Controller (TZC-380) - providing security address region control functions on DDR memory space.
- On-chip RAM (OTCRAM) secure region protection using OTCRAM controller.
- High Assurance Boot (HAB) feature in the System Boot
- Data co-processor (DCP) with SHA-1/SHA-256 hashing algorithms, and AES encryption algorithm with device unique secret key for secure off-chip storage.
- Random number generator (RNGB)

- Secure Non Volatile Storage (SNVS)
- On chip OTP (OCOTP) with on-chip electrical fuses
- Central Security Unit (CSU)
- Secure JTAG Controller (SJC)
- Locked mode in the Smart Direct Memory Access (SDMA) controller

Detailed descriptions of the components are provided in the *Multimedia Applications Processor Security Reference Manual*.

## 11.2 Central Security Unit (CSU)

### 11.2.1 CSU Overview

Security is increasingly important for connected devices such as smart phones, tablets, and integrated media players. Instances of hackers and pirates breaking into portable devices stealing private information and copyright content are now commonplace. As such, an embedded trust architecture is an important feature for many new products. To address security risks and to provide an extensible platform to meet stringent security requirements, this processor incorporates a number of advanced security hardware blocks and architectural features. The Central Security Unit (CSU) is one of those hardware blocks.

The CSU manages the system security policy for peripheral access on the SoC. The CSU allows trusted code to set individual security access privileges on each of the peripherals, using one of eight security access privilege levels. Also, according to programmed policy, the CSU may assign bus master security privileges during bus transactions.

### 11.2.2 CSU Features

The Central Security Unit (CSU) sets access control policies between bus masters and bus slaves, allowing peripherals to be separated into distinct security domains. This protects against unauthorized access to data e.g. when software programs a DMA bus master to access addresses that the software itself is prohibited from accessing directly. Configuring DMA bus master privileges in the CSU consistent with software privileges defends against such attempted accesses.

CSU has the following security related features:

- Peripheral access policy - Appropriate bus master privileges and identity are required to access each peripheral.
- Masters privilege policy - CSU overrides bus master privilege signals, i.e. user/supervisor secure/non-secure, according to access control policy.

### 11.2.3 CSU Functional Description

The CSU enables secure software to set bus privilege security policy within the platform.

Security policies may be set, and optionally locked in the CSU registers, including the command sequence file (CSF) processed by the High Assurance Boot (HAB) or a HABauthenticated image which executes after the boot ROM.

#### 11.2.3.1 CSU Peripheral Access Policy

According to its programmed policy, the CSU determines the bus master privileges and the masters that are allowed to access each of the slave peripherals.

There are four security modes of operation (i.e. bus privileges) in the system distinguished by security (TrustZone/non-TrustZone) and privilege (Supervisor/User) setting of the module. Below is the list of these security modes from the highest security level to the lowest:

- TrustZone (Secure) Privilege (Supervisor) Mode - Highest Security Level
- TrustZone (Secure) non-Privilege (User) Mode - Medium Security Level
- non-TrustZone (Regular) Privilege (Supervisor) Mode - Medium Security Level
- non-TrustZone (Regular) non-Privilege (User) Mode - Lowest Security Level

This functionality is implemented as follows:

The Configure Slave Level (CSL) Register value for a specified peripheral resource defines the output signal -- `csu_sec_level` for that peripheral. The value of this signal determines by what master privileges a peripheral is accessible. The relationship between the value of the `csu_sec_level` signal and security operation mode is shown in the table below. The CSL registers reside in the CSU module. Details, describing CSL register fields and how they are programmed to control access privileges for specific peripherals, can be found in the Security Reference Manual (see System Security Overview section).



**Table 11-1. Permission Access Table**

CSU_SEC_LEVEL[2:0]	Non-Secure User Mode	Non-Secure Spvr Mode	Secure (TZ) User Mode	Secure (TZ) Spvr Mode	CSL register value
(0) 000	RD+WR	RD+WR	RD+WR	RD+WR	8'b1111_1111
(1) 001	None	RD+WR	RD+WR	RD+WR	8'b1011_1011
(2) 010	RD	RD	RD+WR	RD+WR	8'b0011_1111
(3) 011	None	RD	RD+WR	RD+WR	8'b0011_1011
(4) 100	None	None	RD+WR	RD+WR	8'b0011_0011
(5) 101	None	None	None	RD+WR	8'b0010_0010
(6) 110	None	None	RD	RD	8'b0000_0011
(7) 111	None	None	None	None	Any other value

## 11.3 Secure Non-Volatile Storage (SNVS)

### 11.3.1 SNVS Overview

SNVS is a hardware device that includes a security state machine and security violation detection circuits that, together with High Assurance Boot software, determine whether the chip is currently in a secure state.

When the security state machine indicates a secure state, the SNVS allows DCP to use special cryptographic keys to decrypt long-term secrets such as public keypairs, Digital Rights Management keys and proprietary software. When the SNVS detects a potential security violation, such as a tamper alert, the SNVS sends an interrupt to alert the Operating System of the event and denies the use of the special cryptographic keys mentioned above. The SNVS also includes a non-security-related real-time counter.

The SNVS includes the following features:

- Security State Machine driven by High Assurance Boot software and tamper detection circuits
- Master Key Control that protects the integrity and secrecy of the Master Key stored in fuses
- Tamper detection circuits that detect JTAG scan events, power glitches, Master Key ECC check failure, and software-reported and hardware-reported security violations
- Zeroizable Master Key that can be automatically erased in the event of a security breach
- Tamper-protected Secure Realtime Counter that continues running when the chip is powered off

- Non-volatile Monotonic Counter used to protect against “roll-back” attacks
- Non-volatile General Purpose Register can be used to store a 32-bit value across power cycles
- Non-Secure Real Time Counter with programmable alarm and periodic interrupt

### 11.3.2 Tamper Detection

Tamper Detection is a special mechanism provided through a chip pin to help protect the target system “box” against unauthorized opening or tampering.

When not in use, the Tamper Detection signal is pulled-down internally. In case of use, it should be connected to a Tamper Detection contact in a target system (Normally closed, pulled-up to the VDD\_SNVS\_IN).

An always-ON power supply (coin cell) should be present in the system. If the tamper detection feature is enabled by software then opening of the tamper contact:

- Switches system power ON with a Tamper Detection alarm interrupt asserted (for software reaction)
- May activate security related hardware (e.g. erasure of the Zeroizable Master Key and secure memory contents)

## 11.4 Data Co-Processor (DCP)

For security purposes, the Data Co-Processor (DCP) provides a hardware acceleration for cryptographic algorithms. The features of DCP are:

- Encryption Algorithms: AES-128 (ECB and CBC modes)
- Hashing Algorithms: SHA-1 and SHA-256
- Key selection from SNVS, DCP internal key storage or general memory
- Internal Memory for storage of up to four AES-128 keys. Once a key is written to a key slot it may only be read by the DCP AES-128 engine.
- IP slave interface
- DMA

## 11.5 High Assurance Boot (HAB)

>HAB, which is the high assurance boot feature in the system boot ROM, protects the platform from executing unauthorized software (malware) during the boot sequence.

When unauthorized software is permitted to gain control of the boot sequence, it can be exploited for a variety of goals, such as exposing stored secrets; circumventing access controls to sensitive data, services, or networks; or repurposing the platform. Unauthorized software can enter the platform during upgrades or reprovisioning, or when booting from USB connections or removable devices.

HAB protects against unauthorized software by:

- Using digital signatures to recognize authentic software. This allows the user to boot the device to a known initial state, running software signed by the device manufacturer.

## 11.6 System JTAG Controller (SJC)

The JTAG port provides debug access to hardware blocks, including the ARM processor and the system bus. This allows program control and manipulation as well as visibility to the chip peripherals and memory.

The JTAG port must be accessible during initial platform development, manufacturing tests, and general troubleshooting. Given its capabilities, JTAG manipulation is a known attack vector for accessing sensitive data and gaining control over software execution. System JTAG Controller (SJC) protects against the whole range of attacks based on unauthorized JTAG manipulation. It also provides a JTAG port that conforms to IEEE 1149.1 and IEEE 1149.6 (AC) standards for BSR (boundary scan) testing.

SJC provides the following security levels:

- JTAG Disabled-JTAG use is permanently blocked.
- No-Debug-All security sensitive JTAG features are permanently blocked.
- Secure JTAG-JTAG use is restricted (as in the No-Debug level) unless a secret-key challenge/response protocol is successfully executed.
- JTAG Enabled-JTAG use is unrestricted.

Security levels are selected via e-fuse configuration.



## Chapter 12

# ARM Cortex A9 MPCore Platform (ARM)

### 12.1 Overview

The Cortex-A9 Core Platform consists of an ARM<sup>®</sup>Cortex<sup>®</sup>-A9 MPCore processor, which includes a Neon co-processor, a private timer and watch-dog, and 32 KB + 32 KB L1 data and instruction caches per core.

The Cortex-A9 Core Platform consists of a unified 256 KB L2 cache, SCU (Snoop Control Unit), and Generic Interrupt Controller (GIC). In addition, the Cortex-A9 Core Platform includes various components composing the ARM CoreSight debug/Trace system, including PTM and a 2 x CTI, 2xCTM, and 16KB ETB.

The Cortex-A9 processor utilizes two AXI-64 master ports connected from the SCU to the Level 2 Cache. The L2 cache also utilizes 2x AXI-64 to access the L3 memory or other SoC peripherals in a symmetric way.

The core supports debug through real-time trace via PTM, and static debug via JTAG.

The core platform supports static debug through the debug logic to SOC. This includes the capability of real time trace via ARM's CoreSight PTM, ETB and TPIU modules. The CTI and CTM modules allow cross-triggering of internal and external trigger sources.

### 12.2 External Signals

The following table describes the external signals of ARM:

**Table 12-1. ARM External Signals**

Signal	Description	Pad	Mode	Direction
ARM_EVENTI	Input event signal	EPDC_PWRSTAT	ALT4	I
ARM_EVENTO	Output event signal	EPDC_PWRWAKEUP	ALT4	O
ARM_TRACE00	Trace signal	LCD_DAT0	ALT6	O

*Table continues on the next page...*

**Table 12-1. ARM External Signals (continued)**

Signal	Description	Pad	Mode	Direction
ARM_TRACE01	Trace signal	LCD_DAT1	ALT6	O
ARM_TRACE02	Trace signal	LCD_DAT2	ALT6	O
ARM_TRACE03	Trace signal	LCD_DAT3	ALT6	O
ARM_TRACE04	Trace signal	LCD_DAT4	ALT6	O
ARM_TRACE05	Trace signal	LCD_DAT5	ALT6	O
ARM_TRACE06	Trace signal	LCD_DAT6	ALT6	O
ARM_TRACE07	Trace signal	LCD_DAT7	ALT6	O
ARM_TRACE08	Trace signal	LCD_DAT8	ALT6	O
ARM_TRACE09	Trace signal	LCD_DAT9	ALT6	O
ARM_TRACE10	Trace signal	LCD_DAT10	ALT6	O
ARM_TRACE11	Trace signal	LCD_DAT11	ALT6	O
ARM_TRACE12	Trace signal	LCD_DAT12	ALT6	O
ARM_TRACE13	Trace signal	LCD_DAT13	ALT6	O
ARM_TRACE14	Trace signal	LCD_DAT14	ALT6	O
ARM_TRACE15	Trace signal	LCD_DAT15	ALT6	O
ARM_TRACE16	Trace signal	LCD_DAT16	ALT6	O
ARM_TRACE17	Trace signal	LCD_DAT17	ALT6	O
ARM_TRACE18	Trace signal	LCD_DAT18	ALT6	O
ARM_TRACE19	Trace signal	LCD_DAT19	ALT6	O
ARM_TRACE20	Trace signal	LCD_DAT20	ALT6	O
ARM_TRACE21	Trace signal	LCD_DAT21	ALT6	O
ARM_TRACE22	Trace signal	LCD_DAT22	ALT6	O
ARM_TRACE23	Trace signal	LCD_DAT23	ALT6	O
ARM_TRACE24	Trace signal	FEC_RXD0	ALT6	O
ARM_TRACE25	Trace signal	FEC_RX_ER	ALT6	O
ARM_TRACE26	Trace signal	FEC_MDIO	ALT6	O
ARM_TRACE27	Trace signal	FEC_TX_CLK	ALT6	O
ARM_TRACE28	Trace signal	FEC_TX_EN	ALT6	O
ARM_TRACE29	Trace signal	FEC_MDC	ALT6	O

*Table continues on the next page...*

**Table 12-1. ARM External Signals (continued)**

Signal	Description	Pad	Mode	Direction
ARM_TRACE30	Trace signal	FEC_TXD0	ALT6	O
ARM_TRACE31	Trace signal	FEC_CRS_DV	ALT6	O
ARM_TRACE_CLK	Clock signal	LCD_HSYNC	ALT6	O
ARM_TRACE_CTL	Control signal	LCD_VSYNC	ALT6	O

## 12.3 Platform configuration

The [Bus](#), [Cortex A9 Core](#), and [L2 Cache](#) configuration options are contained in the following subsections.

**Table 12-2. Cortex-A9 revision**

Core	MP004-BU-50000-r2p10-0rel0
Neon	AT397-BU-50001- r2p0-00rel0
PL310	PL310-BU-00000-r3p2-00rel0

### 12.3.1 Platform and SCU configuration

**Table 12-3. Cortex-A9 configuration**

Option	Selected Value	Comments
MP_MODE	Yes	Multi-Processor mode
POWER_DOMAIN_WRAPPER	No	Wrappers to support power off of individual cores.
PTM_INTERFACE_PRESENT	Yes	Use PTM as part of Trace/Debug logic.
PARITY	Yes	Using RAM arrays which support parity.
CORE_NUM	1	Number of cores
INT_NUM	128	Number of interrupts (SPIs) in GIC
ACP_PRESENT	No	Accelerator Coherency Port (ACP)
MASTER_NUM	2	Number of 64-bit AXI output master ports.

## 12.3.2 Core configuration

**Table 12-4. Cortex-A9 Core configuration**

Option	Selected Value	Comments
DCACHESIZE	32	L1 Data cache size
ICACHESIZE	32	L1 Instruction cache size
TLBSIZE	128	
JAZELLE_PRESENT	Yes	Providing ARM's Jazelle technology hardware extensions.
FPU_PRESENT	No	The FPU functions are provided by NEON, thus additional FPU cannot be used.
NEON_PRESENT	Yes	Use MPE, NEON Co-Processor and FPU
PRELOAD_ENGINE_PRESENT	No	May only be beneficial in Video processing.

## 12.3.3 PL310 L2 Cache configuration

**Table 12-5. PL310 L2 Cache configuration**

Option	Selected Value	Comments
Cache way size	64 KB	(For total of 256 KB L2 size)
Number of cache ways	16	Performance enhancement versus 8 ways
RAM latencies	4 <sup>1</sup>	
Data RAM banking	Yes	Significantly improves cache throughput
Slave port 1 present	Yes	
Master port 1 present	Yes	
Parity logic	Yes	For military / surveillance applications, and side ease the process of identify memory related issues.
Lockdown by master	Yes	Increase L2 optimization
Lockdown by line	Yes	Increase L2 optimization
AXI ID width	5	
Address filtering	No	Help in timing closure, not required for symmetric AXI bus connectivity scheme.
Speculative read	Yes	Performance boost, when used with CortexA9.
Size of L2 cache	256 KB	Size is implied by Cache-Size times cache-ways (i.e. 256 KB )

1. Preliminary estimate, final value TBD.

## 12.3.4 Endian Modes

The Cortex-A9 Core Platform supports little endian mode only. Big Endian is not supported even though both modes are supported by the Cortex-A9 processor.



### 12.3.5 Memory Parity error support

The i.MX 6SoloLite ARM Cortex A9 MPCore™ platform supports Parity Fail signals for several of the RAM arrays.

Parity fail indication is provided by "PARITYFAILn[7:0]" and "PARITYFAILSCU[N:0]" buses for system notification. On parity error event, these signals will be ORed together to provide a single event (interrupt) to the core, in case of any parity fail event.

## 12.4 Performance and Power

This section will discuss the operational conditions and performance goals for the Cortex-A9 Core Platform.

### 12.4.1 Low-Power design

The Cortex-A9 Core Platform low-power design is based on these characteristics:

- Symmetric processing and ARM design by using the same clock frequency on core
- Low leakage of LP process
- C4 package

As a result, the proposed power modes and power management scheme is as follows:

- The same voltage level must be used for the logic part of the whole platform.
- Separate voltage for memories array is required to allow DVFS.
- On-chip power switches are not utilized by gating the power of the platform directly from the regulator.

#### 12.4.1.1 SRPG (State Retention Power Gating)

ARM core SRPG is implemented by software save & restore of essential configuration registers prior to the complete power-down of the entire ARM platform.

Save & restore utilizes "Dormant mode" for the primary core (L1 cache flushed, L2 preserved), and power down mode of all other cores, as follows:

Power Down flow<sup>2</sup>

---

1. Dormant mode implementation information, provided by ARM, is preliminary.

- Power down request
- Save cores' essential registers and platform registers to L2/DDR memory by Dormant mode routines.
- Perform L1 cache clean operation on all cores
- Enter WFI state
- Power Gating of all cores' and platform logic

#### Power Up flow<sup>2</sup>

- Power up request to GPC external controller (interrupt)
- Supply power
- Reset
- Restore registers of platform and cores from memory by Dormant mode routines.

### 12.4.1.2 Dynamic Voltage and Frequency Scaling (DVFS)

The Cortex-A9 Core Platform has been designed, in conjunction with external control logic and software, to support dynamic scaling of voltage and frequency.

## 12.4.2 Clocks, frequency goals

### 12.4.2.1 ARM Clock

For ARM clock please see the product data sheet.

### 12.4.2.2 Bus Clocks

The AXI master ports are designed to run at half the frequency of the Cortex A9 core clock.

The on-platform debug components clock is asynchronous to the ARM core clock.

### 12.4.2.3 Debug Clocks

The ARM platform contains several debug components.

Their clock frequencies are as follows:

- Trace buffer (ETB) clock - 133 MHz

- Trace port (TPIU) clock - 133 MHz
- AHB clock - 133 MHz

## 12.5 Core Platform Sub-Blocks details

### 12.5.1 ARM Cortex A9 MPCore™ Processor

The information presented in this section focuses on design aspects of the ARM Cortex A9 MPCore™ in the AP subsystem.

The ARM Cortex A9 is a high-performance, low-power, synthesisable processor with an L1 cache subsystem that provides full virtual memory capabilities. The Cortex A9 processor implements the ARMv7-A architecture and runs 32-bit ARM instructions, 16-bit and 32-bit Thumb2 instructions, and 8-bit Java™ byte-codes in Jazelle state.

The ARM Cortex A9 MPCore™ processor in the chip consists of:

- Cortex A9 processors.
- A set of private memory-mapped peripherals, including a global timer and a watchdog and private timer for each Cortex-A9 processor present in the cluster.
- An integrated Interrupt Controller is an implementation of the Generic Interrupt Controller architecture. The integrated Interrupt Controller registers sit beside the timers and watchdog control registers in the private memory region of the Cortex-A9 MPCore.

Individual Cortex-A9 processors in the Cortex-A9 MPCore cluster are symmetrically implemented with hardware configurations as specified in [Platform configuration](#).

### 12.5.2 Media Processing Engine (MPE - NEON)

The Media Processing Engine (MPE) implements ARM NEON technology, a media and signal processing architecture that adds instructions targeted at audio, video, 3-D graphics, image, and speech processing.

Advanced SIMD instructions are available in both ARM and Thumb states. The MPE also implements a VFPv3-D32 Floating-Point Unit.

The ARM Cortex A9 MPCore™ Platform includes MPE per core.

### 12.5.3 Generic Interrupt Controller (GIC)

The Cortex-A9 MPCore contains an integrated interrupt controller that shares the same programmer's model as the PL390 (GIC), although there are implementation-specific differences.

#### 12.5.3.1 Interrupt Controller Features

- 128 interrupt sources
- The Cortex-A9 multiprocessor contains the following types of interrupts:
  - Up to 16 Software Generated Interrupts (SGIs)
  - Private Peripheral Interrupt (PPI) - an interrupt generated by a peripheral that is specific to a single Cortex-A9 processor (there are 5 PPIs for each Cortex-A9 processor interface)
  - Shared Peripheral Interrupt (SPI) - an interrupt generated by a peripheral which the Interrupt Controller can route to any or all Cortex-A9 processor interfaces. The Interrupt Controller supports a maximum of 224 SPIs .
  - Lockable Shared Peripheral Interrupts (LSPI) - there are 31 LSPIs, interrupts 32-62. The user can configure and then lock these interrupts against further change using CFGSDISABLE. The LSPIs are present only if the SPIs are present.

For more information, see ARM MPCORE Technical Reference Manual.

#### 12.5.3.2 About the Interrupt Controller

The Interrupt Controller is a single functional unit that is located in a Cortex-A9 multiprocessor design.

The Interrupt Controller is memory-mapped. The Cortex-A9 processors access it by using a private interface through the SCU.

#### 12.5.3.3 Interrupt Controller Clock frequency

The interrupt controller's clock period is 2x multiple of the main clock period.

The watchdogs and timers use the same clock as the interrupt controller.

### 12.5.3.4 TrustZone support

The Interrupt Controller permits all implemented interrupts to be individually defined as Secure or Non-secure.

The user can program Secure interrupts to use either the IRQ or FIQ interrupt mechanism of a Cortex-A9 processor through the FIQen bit in the ICPICR Register.

Non-secure interrupts are always signalled using the IRQ mechanism of a Cortex-A9 processor.

## 12.5.4 Instruction and data caches (L1)

The Cortex-A9 processor is configured with a 32 Kbyte Instruction Cache and a 32 Kbyte Data Cache.

### 12.5.4.1 L1 features

- Four-way set associative cache
- Virtually indexed and physically addressed
- Capable of providing two words per cycle for all requesting sources
- Eight 32-bit words per cache line
- 128 indexes per tag RAM

## 12.5.5 L2 Cache and controller (PL310)

The platform includes unified (data / instruction) L2 cache unit, based on the cache controller IP by ARM.

The Cortex-A9 processor utilizes 2x AXI-64 master ports connected from the SCU to the Level 2 Cache. The L2 cache also utilizes 2x AXI-64 to access the L3 memory or other SoC peripherals in a symmetric way.

See [Table 12-5](#) for more information.

## 12.6 Debug and Trace Sub-blocks (CoreSight components)

This section gives a brief overview of the modules that are implemented within the Cortex-A9 Core Platform.

The Cortex-A9 Core Platform debug blocks are part of the overall CoreSight debug system which include the 16KB ETB, 2 x CTM's, 2 x CTI's, ATB replicator, DAP, TPIU and APB address decode.

The CoreSight™ compatible Program Flow Trace Macrocell (PTM) provides control for ARM software tracing and debug. The Cross Trigger Interface (CTI) is included in the Cortex-A9 platform to provide a common programming model for use by the debug tools, control the trigger sources, and interface to the Cross Trigger Matrix (CTM). The debug is controlled via an ARM Debug Access Port (DAP).

For details of the full CoreSight debug subsystem, see the [System Debug](#) chapter.

### 12.6.1 Debug Access Port (DAP)

The Debug Access Port (DAP) is an implementation of an ARM Debug Interface version 5.1 (ADIV5.1) comprised of a number of components supplied in a single configuration.

All the supplied components fit into the various architectural components for Debug Ports (DPs), which are used to access the DAP from an external debugger and Access Ports (APs), to access on-chip system resources.

The debug port and access ports together are referred to as the DAP. The DAP provides real-time access for the debugger without halting the processor to:

- AMBA system memory and peripheral registers
- All debug configuration registers

The DAP also provides debugger access to JTAG scan chains of system components, (to non-CoreSight compliant processors, for example).

### 12.6.2 Program Trace Macrocell (PTM)

The PTM unit is a nonintrusive trace macrocell that filters and compresses instruction trace for use in system debugging and system profiling.

The PTM unit has an external interface outside of the processor called the *Advanced Trace Bus* (ATB) interface. The PTM is an evolution of the ETM, designed for the Cortex A9 cores, handling program trace only.

The Cortex A9 PTM provides real time instruction trace for the Cortex A9. It's designed to be used with the CoreSight Design Kit.

Real time tracing is controlled by specifying a set of filtering and triggering resources which include address and data comparators, counters and sequencers.

Two main schemes can be used for connecting PTMs:

1. Single PTM - shared by core and resources.
2. Nx PTMs, where N is numbers of cores in the system.

### 12.6.2.1 Program Flow Trace (PFT)

The CoreSight Program Flow Trace Macrocell (PTM) is based on the Program Flow Trace (PFT) architecture. The PTM generates information that trace tools use to reconstruct the execution of all or part of a program.

The PFT architecture assumes the trace tools can access a copy of the code being traced. For this reason, the PTM generates trace only at certain points in program execution, called waypoints. This reduces the amount of trace data generated by the PTM compared to the ETM protocol. Waypoints are changes in the program flow or events, such as an exception. The trace tools use waypoints to follow the flow of program execution.

For full reconstruction of the program flow, the PTM traces:

- Indirect branches, with target address and condition code
- Direct branches with only the condition code
- Instruction barrier instructions
- Exceptions, with indication of where the exception occurred
- Changes in processor instruction set state
- Changes in processor security state
- Context-ID changes
- Entry to and return from Debug state when Halting Debug-mode is enabled

You can also configure the PTM to trace:

- Cycle count between traced waypoints
- Global system timestamps
- Target addresses for taken direct branches

PTM components include the following main components:

- Processor Interface to monitor the behavior of the processor.
- Trace Generation to create a real-time trace stream.
- Filtering and Triggering Resources used to affect when trace is generated and to control the capturing of trace by the trace tools.
- Main FIFO (72 bytes) flattens out any bursts in the trace stream, and signals an overflow in the trace when it becomes full, halting trace generation until the FIFO empties.
- ATB interface for PTM output.
- APB interface to access the PTM registers.

### 12.6.3 Cross Trigger Interface (CTI)

This block controls the Trigger Interface (TI). The CTI combines and maps the trigger requests, and broadcasts them to all other interfaces on the ECT as channel events.

When the CTI receives a channel event, it maps this onto a trigger output. This enables subsystems to cross trigger with each other. The receiving and transmitting of triggers is performed through the TI. Each CTI has 3/5 trigger inputs.

See [System Debug](#) for more information on the CTI.

### 12.6.4 Embedded Trace Buffer (ETB)

The ETB provides on-chip storage of trace data using 32-bit, 16KB RAM.

The ETB accepts trace data from the Cortex-A9 via an ATB port (passing through a replicator in between). Providing an on-chip buffer alleviates the pin count, bandwidth, and pad design requirements associated with sending trace data to a debugger directly through package pins in near real-time.

Features:

- compiled memory for the trace buffer and can be used as general purpose memory
- AMBA Peripheral Bus programming interface for configuration and memory access

#### 12.6.4.1 AMBA Trace Bus (ATB) Replicator

The ATB Replicator enables two trace sinks (ETB and an off platform port generally connected to a Trace Port Interface Unit-TPIU) to be wired together and receive ATB trace data from the same trace source (PTM).

There are no programmable registers. It takes incoming trace data from a single source (PTM) and replicates it as multiple masters.



## Chapter 13

# AHB to IP Bridge (AIPSTZ)

### 13.1 Overview

This section provides an overview of the AHB to IP Bridge (AIPSTZ). The peripheral bridge acts as an interface between the system bus and lower bandwidth IP Slave (IPS) bus peripherals.

#### 13.1.1 Features

The following list summarizes the key features of the bridge:

- The bridge supports the IPS slave bus signals. This interface is only meant for slave peripherals.
- The bridge supports 8-, 16-, and 32-bit IPS peripherals. (Accesses larger than the size of a peripheral are not supported, except to 32-bit memory.)
- The bridge supports a pair of IPS accesses for 64-bit and certain misaligned AHB transfers to 32-bit memory in 64-bit platforms.
- The bridge directly supports up to 32 16-Kbyte external IPS peripherals, and 2 global external IPS peripheral spaces. The bridge occupies 1 MBytes of total address space.
- The bridge provides configurable per-block and per-master access protections. More details on the protection features and configuration can be found in the Security Reference Manual
- Peripheral read transactions require a minimum of 2 hclk clocks, and unbuffered write transactions require a minimum of 3 hclk clocks.
- The bridge uses one single asynchronous reset and one global clock.

## 13.2 Clocks

The following table describes the clock sources for AIPSTZ. Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 13-1. AIPSTZ Clocks**

Clock name	Clock Root	Description
hclk	ahb_clk_root	Module clock

## 13.3 General Operation

The AHB to IP bridge is the interface between the AHB and on-chip IPS peripherals, which are sub-blocks containing readable/writable control and status registers.

The AHB master reads and writes these registers through the AIPSTZ. The bridge generates block enables, the block address, transfer attributes, byte enables and write data as inputs to the IPS peripherals. The bridge captures read data from the IPS interface and drives it on the AHB.

It occupies a 1-Mbyte portion of the address space. The register maps of the IPS peripherals are located on 16-Kbyte boundaries. Each IPS peripheral is allocated one 16K-byte block of the memory map, and is activated by one of the block enables from the bridge. Up to thirty-two 16-Kbyte external IPS peripherals may be implemented, occupying contiguous blocks of 16 Kbytes. Two global external IPS block enables are available for the remaining address space to allow for customization and expansion of addressed peripheral devices. In addition, a single "non-global" block enable is also asserted whenever any of the thirty-two non-global block enables is asserted.

The bridge is responsible for indicating to IPS peripherals if an access is in supervisor or user mode. It may block user mode accesses to certain IPS peripherals or it may allow the individual IPS peripherals to determine if user mode accesses are allowed. In addition, peripherals may be designated as write-protected.

The bridge supports the notion of "trusted" masters for security purposes. Masters may be individually designated as trusted for reads, trusted for writes, or trusted for both reads and writes, as well as being forced to look as though all accesses from a master are in user-mode privilege level. Refer to [AIPSTZ Memory Map/Register Definition](#) for more information.

All peripheral devices are expected to only require aligned accesses equal to or smaller in size than the peripheral size. An exception to this rule is supported for 32-bit peripherals to allow memory to be placed on the IPS.

## 13.4 Functional Description

The AIPS bridge serves as a protocol translator between the AHB system bus and the IP bus.

Support is provided for generating a pair of 32-bit IP bus accesses when targeted by a 64-bit system bus access, or a misaligned access which crosses a 32-bit boundary. No other bus-sizing access support is provided.

## 13.5 Access Protections

The AIPSTZ bridge provides programmable access protections for both masters and peripherals. It allows the privilege level of a master to be overridden, forcing it to user-mode privilege, and allows masters to be designated as trusted or untrusted.

Peripherals may require supervisor privilege level for access, may restrict access to a trusted master only, and may be write-protected. IP bus peripherals are subject to access control policies set in both CSU registers and AIPSTZ registers. An access is blocked if it is denied by either policy.

## 13.6 Access Support

Aligned 64-bit accesses, aligned and misaligned word and half word accesses, as well as byte accesses are supported for 32-bit peripherals. Misaligned accesses are supported to allow memory to be placed on the IPS.

Peripheral registers must not be misaligned, although no explicit checking is performed by the AIPS bridge. The bridge will perform two IPS transfers for 64-bit accesses, word accesses with byte offsets of 1, 2, or 3, and for half word accesses with a byte offset of 3. All other accesses will be performed with a single IPS transfer.

Only aligned half word and byte accesses are supported for 16-bit peripherals. All other accesses types are unsupported, and results of such accesses are undefined. They are not terminated with an error response.

Only byte accesses are supported for 8-bit peripherals. All other accesses types are unsupported, and results of such accesses are undefined. They are not terminated with an error response.

## 13.7 Initialization Information

The AIPS bridge should be programmed before use.

The following registers should be initialized: The Master Privilege Registers (AIPSTZ\_MPRs), the Peripheral Access Control registers (AIPSTZ\_PACRs), and the Off-platform Peripheral Access Control registers (AIPSTZ\_OPACRs) described in [AIPSTZ Memory Map/Register Definition](#).

### 13.7.1 Security Block

The AIPSTZ contains a security block that is connected to each off-platform peripheral. This block filters accesses based on write/read, non-secure, and supervisor signals.

Each peripheral can be individually configured to allow or deny each of the following transactions as described in the table below:

**Table 13-2. Peripheral Access Configuration options**

Config Bit	Write	Non-Secure	Supervisor	Meaning
0	0	0	0	Secure User Read
1	0	0	1	Secure Supervisor Read
2	0	1	0	Non-Secure User Read
3	0	1	1	Non-Secure Supervisor Read
4	1	0	0	Secure User Write
5	1	0	1	Secure Supervisor Write
6	1	1	0	Non-Secure User Write
7	1	1	1	Non-Secure Supervisor Write

Each peripheral has a security configuration (sec\_config\_X) input for determining whether to allow or deny a given access type. These are 8-bit vectors, with each bit corresponding to one of the transactions above as listed in the Config Bit column of [Table 13-2](#). If the bit is asserted (1'b1), the transaction is allowed. If the bit is negated (1'b0), the transaction is not allowed.

For example, if peripheral 0 is configured as follows:

sec\_config\_0 [7:0] = 8'b0011\_0011

This peripheral can only be accessed by secure transactions. Bits 0, 1, 4, and 5 are asserted and these bits refer to the four types of secure transactions. If an insecure transaction is attempted to this peripheral, it will result in an error.

Eight bits per peripheral across an entire system can result in a large number of configuration bits that must be assigned and controlled, most likely in a series of registers in another block. To reduce the number of register bits required predefined sets of security profiles can be defined and encapsulated in an external security translation block. The table below describes one set of security profiles that has been proposed for use with the AIPSTZ.

**Table 13-3. Security Levels**

CSU_SEC_LEVEL	Non-Secure User	Non-Secure Supervisor	Secure User	Secure Supervisor
0	RD+WR	RD+WR	RD+WR	RD+WR
1	NOT ALLOWED	RD+WR	RD+WR	RD+WR
2	Read Only	Read Only	RD+WR	RD+WR
3	NOT ALLOWED	Read Only	RD+WR	RD+WR
4	NOT ALLOWED	NOT ALLOWED	RD+WR	RD+WR
5	NOT ALLOWED	NOT ALLOWED	NOT ALLOWED	RD+WR
6	NOT ALLOWED	NOT ALLOWED	Read Only	Read Only
7	NOT ALLOWED	NOT ALLOWED	NOT ALLOWED	NOT ALLOWED

Information regarding CSU is provided in the Security Reference Manual. Contact your Freescale representative for information about obtaining this document.

A 3-bit input, 8-bit output translation block can be used such that only three register bits are required to set the security profile and the translation block will drive the correct 8-bit configuration vector. Each peripheral connected to the AIPSTZ would require this translation block. The top level AIPSTZ has this three bit input line `csu\_sec\_level[2:0]' corresponding to each peripheral X.

## 13.8 AIPSTZ Memory Map/Register Definition

The memory map for the AIPS SW-visible registers is shown in the table below.

The MPROT and OPACR fields are 4 bits in width. Some bits may be reserved depending on device.

## AIPSTZ memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
200_0000	Master Priviledge Registers (AIPSTZ1_MPR)	32	R/W	0496_ED40h	<a href="#">13.8.1/426</a>
200_0040	Off-Platform Peripheral Access Control Registers (AIPSTZ1_OPACR)	32	R/W	4444_4444h	<a href="#">13.8.2/428</a>
200_0044	Off-Platform Peripheral Access Control Registers (AIPSTZ1_OPACR1)	32	R/W	4444_4444h	<a href="#">13.8.3/431</a>
200_0048	Off-Platform Peripheral Access Control Registers (AIPSTZ1_OPACR2)	32	R/W	4444_4444h	<a href="#">13.8.4/434</a>
200_004C	Off-Platform Peripheral Access Control Registers (AIPSTZ1_OPACR3)	32	R/W	4444_4444h	<a href="#">13.8.5/437</a>
200_0050	Off-Platform Peripheral Access Control Registers (AIPSTZ1_OPACR4)	32	R/W	4444_4444h	<a href="#">13.8.6/440</a>
210_0000	Master Priviledge Registers (AIPSTZ2_MPR)	32	R/W	0496_ED40h	<a href="#">13.8.1/426</a>
210_0040	Off-Platform Peripheral Access Control Registers (AIPSTZ2_OPACR)	32	R/W	4444_4444h	<a href="#">13.8.2/428</a>
210_0044	Off-Platform Peripheral Access Control Registers (AIPSTZ2_OPACR1)	32	R/W	4444_4444h	<a href="#">13.8.3/431</a>
210_0048	Off-Platform Peripheral Access Control Registers (AIPSTZ2_OPACR2)	32	R/W	4444_4444h	<a href="#">13.8.4/434</a>
210_004C	Off-Platform Peripheral Access Control Registers (AIPSTZ2_OPACR3)	32	R/W	4444_4444h	<a href="#">13.8.5/437</a>
210_0050	Off-Platform Peripheral Access Control Registers (AIPSTZ2_OPACR4)	32	R/W	4444_4444h	<a href="#">13.8.6/440</a>

## 13.8.2 Master Priviledge Registers (AIPSTZx\_MPR)

Each AIPSTZ\_MPR specifies 16 4-bit fields defining the access privilege level associated with a bus master in the platform, as well as specifying whether write accesses from this master are bufferable shown in [Table 13-6](#)

The registers provide one field per bus master, where field 15 corresponds to master 15, field 14 to master 14,... field 0 to master 0 (typically the processor core). The master index allocation is shown in [Table 13-7](#).

**Table 13-16. MPROT Field**

Bit	Field	Description
3	MBW	<b>Master Buffer Writes</b> - This bit determines whether the AIPSTZ is enabled to buffer writes from this master.
2	MTR	<b>Master Trusted for Reads</b> - This bit determines whether the master is trusted for read accesses.
1	MTW	<b>Master Trusted for Writes</b> - This bit determines whether the master is trusted for write accesses.

*Table continues on the next page...*

**Table 13-16. MPROT Field (continued)**

Bit	Field	Description
0	MPL	<b>Master Privilege Level</b> - This bit determines how the privilege level of the master is determined.

**NOTE**

The reset value is set to 0000\_0000\_7700\_0000, which makes master 0 and master 1 (ARM CORE) the trusted masters. Trusted software can change the settings after reset.

**Table 13-17. Master Index Allocation**

Master Index	Master Name	Comments
Master 0	All masters excluding ARM core and SDMA	Share the same number allocation.
Master 1	ARM CORE	
Master 2	Reserved	
Master 3	SDMA	
Master 4-15	Reserved	

Address: Base address + 0h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R									Reserved				MPROT3				Reserved															
W	MPROT0				MPROT1				Reserved				MPROT3				Reserved															
Reset	0	0	0	0	0	1	0	0	1	0	0	1	0	1	1	0	1	1	1	0	1	1	0	1	0	1	0	0	0	0	0	0

**AIPSTZx\_MPR field descriptions**

Field	Description
31–28 MPROT0	<p>Master 0 Privilege, Buffer, Read, Write Control</p> <p>xxx0 <b>MPL</b> — Accesses from this master are forced to user-mode (ips_supervisor_access is forced to zero) regardless of the hprot[1] access attribute.</p> <p>xxx1 <b>MPL</b> — Accesses from this master are not forced to user-mode. The hprot[1] access attribute is used directly to determine ips_supervisor_access.</p> <p>xx0x <b>MTW</b> — This master is not trusted for write accesses.</p> <p>xx1x <b>MTW</b> — This master is trusted for write accesses.</p> <p>x0xx <b>MTR</b> — This master is not trusted for read accesses.</p> <p>x1xx <b>MTR</b> — This master is trusted for read accesses.</p> <p>0xxx <b>MBW</b> — Write accesses from this master are not bufferable</p> <p>1xxx <b>MBW</b> — Write accesses from this master are allowed to be buffered</p>
27–24 MPROT1	<p>Master 1 Privilege, Buffer, Read, Write Control</p> <p>xxx0 <b>MPL</b> — Accesses from this master are forced to user-mode (ips_supervisor_access is forced to zero) regardless of the hprot[1] access attribute.</p> <p>xxx1 <b>MPL</b> — Accesses from this master are not forced to user-mode. The hprot[1] access attribute is used directly to determine ips_supervisor_access.</p> <p>xx0x <b>MTW</b> — This master is not trusted for write accesses.</p>

*Table continues on the next page...*

**AIPSTZx\_MPR field descriptions (continued)**

Field	Description
xx1x x0xx x1xx 0xxx 1xxx	<b>MTW</b> — This master is trusted for write accesses. <b>MTR</b> — This master is not trusted for read accesses. <b>MTR</b> — This master is trusted for read accesses. <b>MBW</b> — Write accesses from this master are not bufferable <b>MBW</b> — Write accesses from this master are allowed to be buffered
23–20 -	This field is reserved. Reserved
19–16 MPROT3	Master 3 Privilege, Buffer, Read, Write Control.  xxx0 <b>MPL</b> — Accesses from this master are forced to user-mode (ips_supervisor_access is forced to zero) regardless of the hprot[1] access attribute. xxx1 <b>MPL</b> — Accesses from this master are not forced to user-mode. The hprot[1] access attribute is used directly to determine ips_supervisor_access. xx0x <b>MTW</b> — This master is not trusted for write accesses. xx1x <b>MTW</b> — This master is trusted for write accesses. x0xx <b>MTR</b> — This master is not trusted for read accesses. x1xx <b>MTR</b> — This master is trusted for read accesses. 0xxx <b>MBW</b> — Write accesses from this master are not bufferable 1xxx <b>MBW</b> — Write accesses from this master are allowed to be buffered
15–0 -	This field is reserved. Reserved

### 13.8.3 Off-Platform Peripheral Access Control Registers (AIPSTZx\_OPACR)

Each of the off-platform peripherals have an Off-platform Peripheral Access Control Register (AIPSTZ\_OPACR) which defines the access levels supported by the given block.

Each AIPSTZ\_OPACR has the following format shown in [Table 13-9](#)

**Table 13-19. OPAC Field**

Bit	Field	Description
3	BW	<b>Buffer Writes</b> - This bit determines whether write accesses to this peripheral are allowed to be buffered. <sup>1</sup>
2	SP	<b>Supervisor Protect</b> - This bit determines whether the peripheral requires supervisor privilege level for access.
1	WP	<b>Write Protect</b> - This bit determines whether the peripheral allows write accesses.
0	TP	<b>Trusted Protect</b> - This bit determines whether the peripheral allows accesses from an untrusted master.

1. Buffered writes are not available for AIPSTZ. This bit should be set to '0'.
1. Buffered writes are not available for AIPSTZ. This bit should be set to '0'.
1. Buffered writes are not available for AIPSTZ. This bit should be set to '0'.



Address: Base address + 40h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																
R	OPAC0								OPAC1								OPAC2								OPAC3								OPAC4								OPAC5								OPAC6								OPAC7							
W																																																																
Reset	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0																																

## AIPSTZx\_OPACR field descriptions

Field	Description
31–28 OPAC0	<p>Off-platform Peripheral Access Control 0</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
27–24 OPAC1	<p>Off-platform Peripheral Access Control 1</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
23–20 OPAC2	<p>Off-platform Peripheral Access Control 2</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p>

Table continues on the next page...

## AIPSTZx\_OPACR field descriptions (continued)

Field	Description
	<p>master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
19–16 OPAC3	<p>Off-platform Peripheral Access Control 3</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
15–12 OPAC4	<p>Off-platform Peripheral Access Control 4</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
11–8 OPAC5	<p>Off-platform Peripheral Access Control 5</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p>

*Table continues on the next page...*

**AIPSTZx\_OPACR field descriptions (continued)**

Field	Description
	0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
7–4 OPAC6	Off-platform Peripheral Access Control 6  xxx0 <b>TP</b> — Accesses from an untrusted master are allowed. xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x <b>WP</b> — This peripheral allows write accesses. xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses. x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
3–0 OPAC7	Off-platform Peripheral Access Control 7  xxx0 <b>TP</b> — Accesses from an untrusted master are allowed. xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x <b>WP</b> — This peripheral allows write accesses. xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses. x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.

1. Buffered writes are not available for AIPSTZ. This bit should be set to '0'.

### 13.8.4 Off-Platform Peripheral Access Control Registers (AIPSTZx\_OPACR1)

Each of the off-platform peripherals have an Off-platform Peripheral Access Control Register (AIPSTZ\_OPACR) which defines the access levels supported by the given block.

Each AIPSTZ\_OPACR has the following format shown in [Table 13-9](#)

## AIPSTZ Memory Map/Register Definition

Address: Base address + 44h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																
R	OPAC8								OPAC9								OPAC10								OPAC11								OPAC12								OPAC13								OPAC14								OPAC15							
W																																																																
Reset	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0																																

### AIPSTZx\_OPACR1 field descriptions

Field	Description
31–28 OPAC8	<p>Off-platform Peripheral Access Control 8</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
27–24 OPAC9	<p>Off-platform Peripheral Access Control 9</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
23–20 OPAC10	<p>Off-platform Peripheral Access Control 10</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p>

Table continues on the next page...

**AIPSTZx\_OPACR1 field descriptions (continued)**

Field	Description
	<p>master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
19–16 OPAC11	<p>Off-platform Peripheral Access Control 11</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
15–12 OPAC12	<p>Off-platform Peripheral Access Control 12</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
11–8 OPAC13	<p>Off-platform Peripheral Access Control 13</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p>

*Table continues on the next page...*

**AIPSTZx\_OPACR1 field descriptions (continued)**

Field	Description
	0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
7–4 OPAC14	Off-platform Peripheral Access Control 14  xxx0 <b>TP</b> — Accesses from an untrusted master are allowed. xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x <b>WP</b> — This peripheral allows write accesses. xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses. x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
3–0 OPAC15	Off-platform Peripheral Access Control 15  xxx0 <b>TP</b> — Accesses from an untrusted master are allowed. xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x <b>WP</b> — This peripheral allows write accesses. xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses. x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.

### 13.8.5 Off-Platform Peripheral Access Control Registers (AIPSTZx\_OPACR2)

Each of the off-platform peripherals have an Off-platform Peripheral Access Control Register (AIPSTZ\_OPACR) which defines the access levels supported by the given block.

Each AIPSTZ\_OPACR has the following format shown in [Table 13-9](#)

Address: Base address + 48h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																
R	OPAC16								OPAC17								OPAC18								OPAC19								OPAC20								OPAC21								OPAC22								OPAC23							
W																																																																
Reset	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0																																

## AIPSTZx\_OPACR2 field descriptions

Field	Description
31–28 OPAC16	<p>Off-platform Peripheral Access Control 16</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
27–24 OPAC17	<p>Off-platform Peripheral Access Control 17</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
23–20 OPAC18	<p>Off-platform Peripheral Access Control 18</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p>

Table continues on the next page...

## AIPSTZx\_OPACR2 field descriptions (continued)

Field	Description
	<p>master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
19–16 OPAC19	<p>Off-platform Peripheral Access Control 19</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
15–12 OPAC20	<p>Off-platform Peripheral Access Control 20</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
11–8 OPAC21	<p>Off-platform Peripheral Access Control 21</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p>

*Table continues on the next page...*



**AIPSTZx\_OPACR2 field descriptions (continued)**

Field	Description
	0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
7–4 OPAC22	Off-platform Peripheral Access Control 22  xxx0 <b>TP</b> — Accesses from an untrusted master are allowed. xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x <b>WP</b> — This peripheral allows write accesses. xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses. x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
3–0 OPAC23	Off-platform Peripheral Access Control 23  xxx0 <b>TP</b> — Accesses from an untrusted master are allowed. xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x <b>WP</b> — This peripheral allows write accesses. xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses. x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.

### 13.8.6 Off-Platform Peripheral Access Control Registers (AIPSTZx\_OPACR3)

Each of the off-platform peripherals have an Off-platform Peripheral Access Control Register (AIPSTZ\_OPACR) which defines the access levels supported by the given block.

Each AIPSTZ\_OPACR has the following format shown in [Table 13-9](#)

## AIPSTZ Memory Map/Register Definition

Address: Base address + 4Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W	OPAC24				OPAC25				OPAC26				OPAC27				OPAC28				OPAC29				OPAC30				OPAC31			
Reset	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0

### AIPSTZx\_OPACR3 field descriptions

Field	Description
31–28 OPAC24	<p>Off-platform Peripheral Access Control 24</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
27–24 OPAC25	<p>Off-platform Peripheral Access Control 25</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
23–20 OPAC26	<p>Off-platform Peripheral Access Control 26</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p>

Table continues on the next page...

**AIPSTZx\_OPACR3 field descriptions (continued)**

Field	Description
	<p>master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
19–16 OPAC27	<p>Off-platform Peripheral Access Control 27</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
15–12 OPAC28	<p>Off-platform Peripheral Access Control 28</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
11–8 OPAC29	<p>Off-platform Peripheral Access Control 29</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p>

*Table continues on the next page...*

**AIPSTZx\_OPACR3 field descriptions (continued)**

Field	Description
	0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
7–4 OPAC30	Off-platform Peripheral Access Control 30  xxx0 <b>TP</b> — Accesses from an untrusted master are allowed. xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x <b>WP</b> — This peripheral allows write accesses. xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses. x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.
3–0 OPAC31	Off-platform Peripheral Access Control 31  xxx0 <b>TP</b> — Accesses from an untrusted master are allowed. xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. xx0x <b>WP</b> — This peripheral allows write accesses. xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses. x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus. 0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ. 1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.

### 13.8.7 Off-Platform Peripheral Access Control Registers (AIPSTZx\_OPACR4)

Each of the off-platform peripherals have an Off-platform Peripheral Access Control Register (AIPSTZ\_OPACR) which defines the access levels supported by the given block.

Each AIPSTZ\_OPACR has the following format shown in [Table 13-9](#)

Address: Base address + 50h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OPAC32								OPAC33								Reserved															
W	OPAC32								OPAC33								Reserved															
Reset	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0

## AIPSTZx\_OPACR4 field descriptions

Field	Description
31–28 OPAC32	<p>Off-platform Peripheral Access Control 32</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
27–24 OPAC33	<p>Off-platform Peripheral Access Control 33</p> <p>xxx0 <b>TP</b> — Accesses from an untrusted master are allowed.</p> <p>xxx1 <b>TP</b> — Accesses from an untrusted master are not allowed. If an access is attempted by an untrusted master, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>xx0x <b>WP</b> — This peripheral allows write accesses.</p> <p>xx1x <b>WP</b> — This peripheral is write protected. If a write access is attempted, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>x0xx <b>SP</b> — This peripheral does not require supervisor privilege level for accesses.</p> <p>x1xx <b>SP</b> — This peripheral requires supervisor privilege level for accesses. The master privilege level must indicate supervisor via the hprot[1] access attribute, and the MPROTx[MPL] control bit for the master must be set. If not, the access is terminated with an error response and no peripheral access is initiated on the IPS bus.</p> <p>0xxx <b>BW</b> — Write accesses to this peripheral are not bufferable by the AIPSTZ.</p> <p>1xxx <b>BW</b> — Write accesses to this peripheral are allowed to be buffered by the AIPSTZ.</p>
23–0 -	<p>This field is reserved.</p> <p>Reserved</p>



# Chapter 14

## Digital Audio Multiplexer (AUDMUX)

### 14.1 Overview

The Digital Audio Multiplexer (AUDMUX) provides a programmable interconnect device for voice, audio, and synchronous data routing between Synchronous Serial Interface Controller (SSI) and audio/voice codec's (also known as coder-decoders) peripheral serial interfaces.

This section includes a top level diagram that shows the functional organization of the block, including all off-chip signals.

AUDMUX allows the users to reconfigure the audio system signal routing through programming, as opposed to altering the PCB design. The full description of the block is in [Functional Description](#).

[Figure 14-1](#) shows the block diagram.

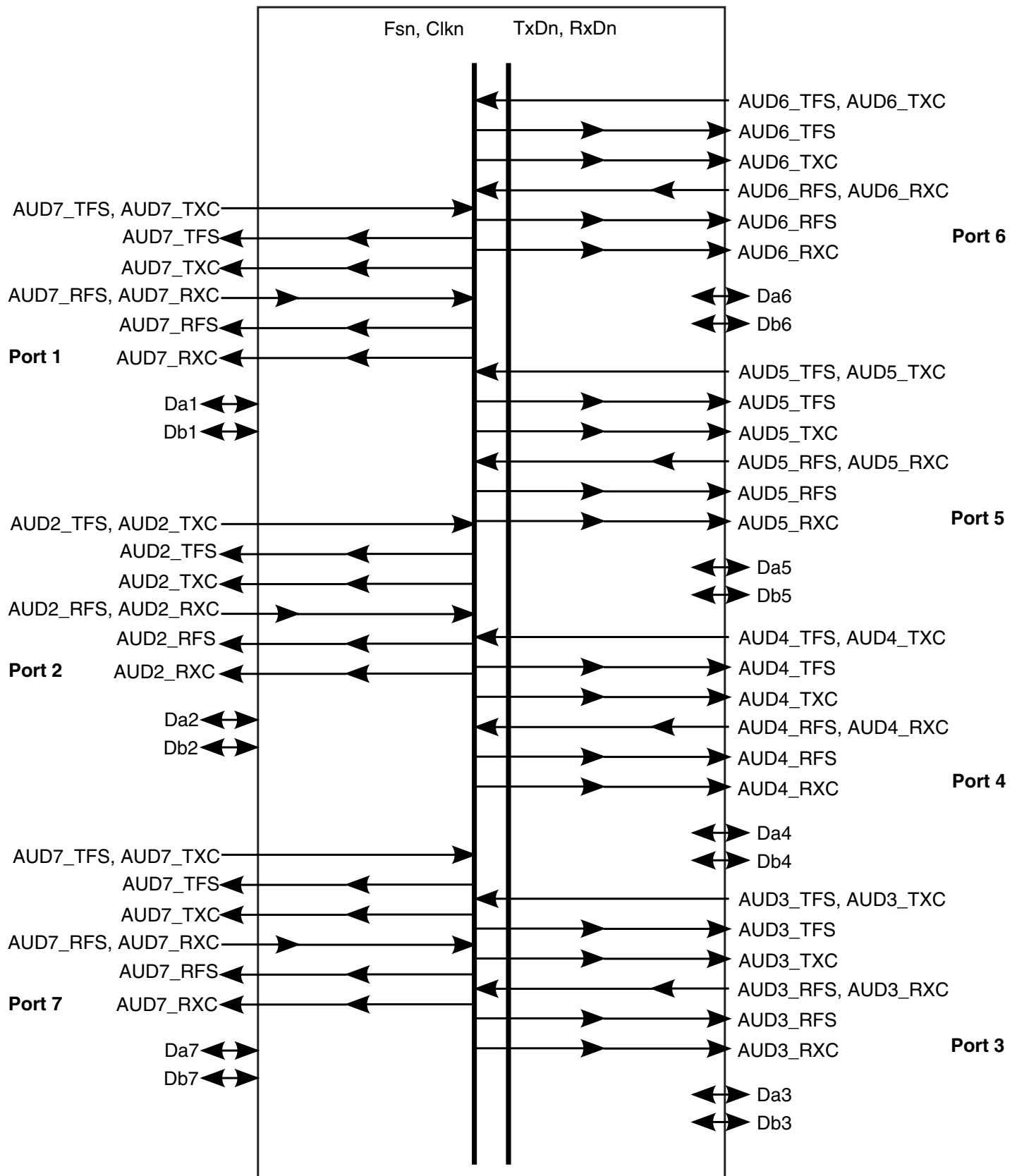


Figure 14-1. AUDMUX Block Diagram



AUDMUX supports single source to single destination connectivity or single source to multiple destination connectivity.

AUDMUX includes two types of interfaces: Internal ports and External ports. Internal ports are hard wired to Synchronous Serial Interface Controller (SSI). The connection between each SSI and AUDMUX's Internal ports cannot be modified, however, routing of the signals connected to each of the internal port can be routed within AUDMUX. External ports are connected to IOMUX module where the ports connect to off-chip audio devices and serial interfaces of other processors. The connectivity of the External port and IOMUX cannot be configured, but the output or input of the signal can be routed easily by setting the appropriate AUDMUX registers.

### 14.1.1 Features

Key features of the block include:

- Two or Three internal ports (depends on number of SSI supported on the device)
- Four external ports
- Full 6-wire SSI interfaces for asynchronous receive and transmit
- Configurable 4-wire (synchronous) or 6-wire (asynchronous) peripheral interfaces
- Independent Tx/Rx Frame sync and clock direction selection for host or peripheral
- Each host interface's capability to connect to any other host or peripheral interface in a point-to-point or point-to-multipoint (network mode)

### 14.1.2 Modes and Operations

The AUDMUX supports the modes described in [Operating Modes](#).

## 14.2 External Signals

The following table describes the external signals of AUDMUX:

**Table 14-1. AUDMUX External Signals**

Signal	Description	Pad	Mode	Direction
AUD3_RXC (RXC)	Receive clock signal	AUD_RXC	ALT0	IO
AUD3_RXD (RXD)	Data receive signal	AUD_RXD	ALT0	IO
AUD3_RXFS (RXFS)	Receive Frame sync signal	AUD_RXFS	ALT0	IO
AUD3_TXC (TXC)	Transmit clock signal	AUD_TXC	ALT0	IO

*Table continues on the next page...*

**Table 14-1. AUDMUX External Signals (continued)**

Signal	Description	Pad	Mode	Direction
AUD3_TXD (TXD)	Data transmit signal	AUD_TXD	ALT0	IO
AUD3_TXFS (TXFS)	Transmit Frame sync signal	AUD_TXFS	ALT0	IO
AUD4_RXC (RXC)	Receive clock signal	I2C2_SDA	ALT1	IO
		LCD_DAT2	ALT4	
		SD2_CMD	ALT1	
AUD4_RXD (RXD)	Data receive signal	ECSP11_SS0	ALT1	IO
		LCD_DAT3	ALT4	
		SD2_DAT0	ALT1	
AUD4_RXFS (RXFS)	Receive Frame sync signal	I2C2_SCL	ALT1	IO
		LCD_DAT1	ALT4	
		SD2_CLK	ALT1	
AUD4_TXC (TXC)	Transmit clock signal	ECSP11_MOSI	ALT1	IO
		LCD_DAT4	ALT4	
		SD2_DAT1	ALT1	
AUD4_TXD (TXD)	Data transmit signal	ECSP11_SCLK	ALT1	IO
		LCD_DAT6	ALT4	
		SD2_DAT3	ALT1	
AUD4_TXFS (TXFS)	Transmit Frame sync signal	ECSP11_MISO	ALT1	IO
		LCD_DAT5	ALT4	
		SD2_DAT2	ALT1	
AUD5_RXC (RXC)	Receive clock signal	EPDC_PWRCTRL0	ALT1	IO
		SD3_CMD	ALT1	
AUD5_RXD (RXD)	Data receive signal	SD3_DAT0	ALT1	IO
AUD5_RXFS (RXFS)	Receive Frame sync signal	EPDC_VCOM0	ALT1	IO
		SD3_CLK	ALT1	
AUD5_TXC (TXC)	Transmit clock signal	SD3_DAT1	ALT1	IO
AUD5_TXD (TXD)	Data transmit signal	EPDC_PWRCTRL2	ALT1	IO
		SD3_DAT3	ALT1	
AUD5_TXFS (TXFS)	Transmit Frame sync signal	EPDC_PWRCTRL1	ALT1	IO
		SD3_DAT2	ALT1	
AUD6_RXC (RXC)	Receive clock signal	FEC_TX_CLK	ALT2	IO
		KEY_ROW3	ALT1	
AUD6_RXD (RXD)	Data receive signal	FEC_RX_ER	ALT2	IO
		KEY_COL4	ALT1	
AUD6_RXFS (RXFS)	Receive Frame sync signal	FEC_MDIO	ALT2	IO
		KEY_COL3	ALT1	
AUD6_TXC (TXC)	Transmit clock signal	FEC_CRD_DV	ALT2	IO
		KEY_ROW4	ALT1	

Table continues on the next page...

**Table 14-1. AUDMUX External Signals (continued)**

Signal	Description	Pad	Mode	Direction
AUD6_TXD (TXD)	Data transmit signal	FEC_TXD0	ALT2	IO
		KEY_ROW5	ALT1	
AUD6_TXFS (TXFS)	Transmit Frame sync signal	FEC_RXD1	ALT2	IO
		KEY_COL5	ALT1	
AUDIO_CLK_OUT (CLK_OUT)	Clock output signal	AUD_MCLK	ALT0	IO
		FEC_MDC	ALT2	
		LCD_DAT7	ALT4	
		PWM1	ALT2	

## 14.3 Clocks

This section provides information about AUDMUX clocking including clock inputs and the clock diagram.

The following table describes the clock source for AUDMUX. Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 14-2. AUDMUX Clocks**

Clock name	Clock Root	Description
ipg_clk_s	ipg_clk_root	Peripheral access clock

### 14.3.1 Clock Inputs

The IP Bus read/write clock-peripheral clock (ipg\_clk\_s) is an input to the AUDMUX. It is used for all AUDMUX register accesses. It is driven only when there is an AUDMUX access on the IP Bus.

### 14.3.2 Clock Diagram

The figure below shows the clocking used in the AUDMUX.

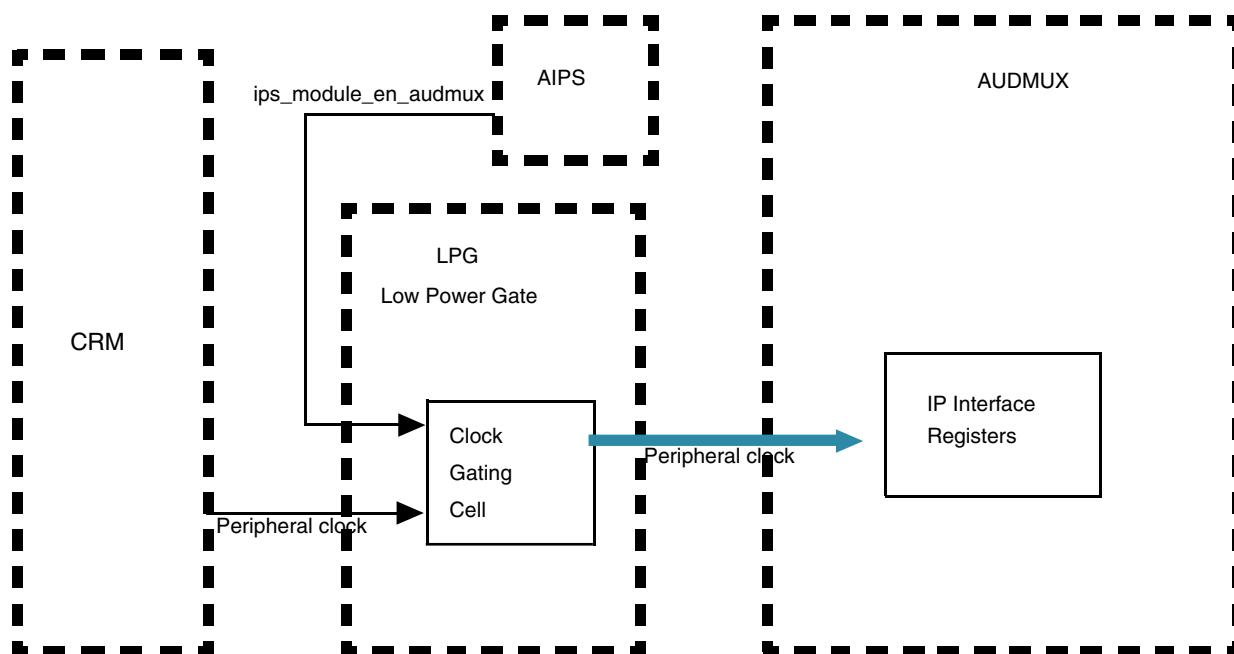


Figure 14-2. AUDMUX Clocking Scheme

### 14.3.3 Clocking Restrictions

- Since the AUDMUX requires only peripheral clock, it places no restrictions on the bus frequency.
- All registers in the AUDMUX are control registers so their values will not change frequently. These values will be programmed when changing between use cases (not during operation in a particular mode).

## 14.4 Default Register Configuration

There are two configuration registers for each port. Each pair of configuration registers is identical for each port; however, the default values following a reset differ as shown in the Memory Map.

[Default Port Configuration](#), describes the default configuration of the ports.

### 14.4.1 Default Port Configuration

After a reset, each port defaults to normal mode ( $PDCRn[MODE] = 0$ ) with synchronous timing mode ( $PTCRn[SYN] = 1$ ) enabled.

The default port-to-port connections are as follows:

- Port 1 to Port 6
  - Port 6 provides the clock and frame sync.
- Port 2 to Port 5
  - Port 5 provides the clock and frame sync.
- Port 3 to Port 4
  - Port 4 provides the clock and frame sync.
- Port 7 to Port 7 (in data loopback mode)
  - Clock and frame syncs are inputs.

## 14.5 Functional Description

This section provides a complete functional description of the AUDMUX.

### 14.5.1 Operating Modes

This section describes all functional operation modes of the AUDMUX.

Figure 14-1 shows the AUDMUX block diagram.

All of the ports are essentially identical; there is no functional difference among Ports 1 through 7. The main difference is whether a port is hard wired to synchronous serial interface (SSI) or hard wired to the chip's pads. Each of the connection is hard wired to specific AUDMUX port. AUDMUX provides flexibility in routing the signal within the module, but all Internal and External port connections are fixed to specific configuration.

All ports can be configured as four- or six-wire interfaces. When configured as a six-wire interface, Receive Frame Sync (RXFS) and Receive Clock (RXC) signals of SSI interface enable the serial interface to be used in asynchronous mode with separate receive and transmit clocks.

AUDMUX supports both Normal mode (not to be confused with SSI's Normal Mode), External Network mode and Internal Network mode. The definition of each mode will be given in the next section.

All ports have a TXRXEN bit to provide flexibility in supporting network mode configurations. The TXRXEN bit reverses the functions of transmit and receive data lines where the transmit line is configured as receive and transmit line to be configured as transmit line. This function is provided so that mastership of the serial bus can be passed among multiple external devices connected to a single port.

In addition to supporting the External Network mode (default), all ports support an internal network mode:

- With internal network mode, single point-to-multipoint network configuration with an arbitrary number of slaves can be supported if the external slaves are put into the high-impedance state (as defined in the SSI network mode protocol) and have pull-up resistors on their TxD pins. (Alternatively, this can be viewed as requiring a pull-up resistor on the corresponding AUDMUX RxD pin.)

Bit clock direction selection enables each port to be configured as a master or slave in the flow.

Possible scenarios include:

- SSI (hard wired to internal port) transmits data to a voice codec and a BT (Blue tooth) codec (both on external Port 4Port 5) and the Bottom Connector (on external Port 5Port 6) simultaneously using network mode. SSI is configured as the master.
- An external processor (external port - Port 4Port 3) drives a voice codec and a BT codec (both on external Port 5Port 4) and the Bottom Connector (on Port 6Port 5) simultaneously using network mode. The external processor is the master.

### 14.5.1.1 Port Receive Data Modes

Each port has logic to select which data lines are used to create the RXD line for the corresponding host interface.

Figure 14-3 shows the logic used to create the RXD line for Port 1. This logic has the following modes of operation (as determined by MODE:

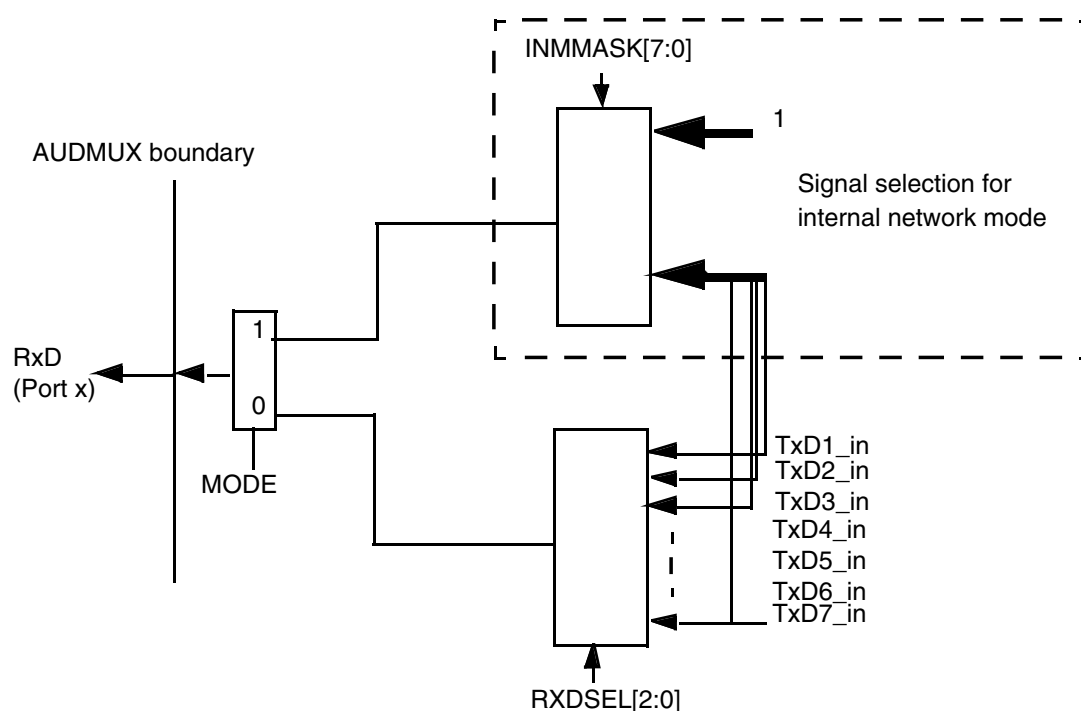
- Normal (not to be confused with SSI's normal mode)
- Internal network mode

The subsequent sections describe the various modes of the port receive data logic. The following terms are used to define the operation of the AUDMUX:

- Network mode- Time-Division Multiplexed protocol for sending unique data to multiple devices on a serial bus or single devices with multi-channel capabilities.
- Internal network mode-Physical bus configuration where multiple serial buses are effectively connected within the AUDMUX via digital logic to create point-to

multipoint connectivity. An arbitrary number of devices are supported. Devices must be put into the high-impedance state as specified by the network mode protocol.

- External network mode-Physical bus configuration where multiple serial buses are electrically connected together on a printed circuit board (that is, external to the AUDMUX). Devices must put their TXD lines into the high-impedance state as specified by the network mode protocol. TXD lines of devices must be pulled high.



**Figure 14-3. Receive Data Logic for Port x**

#### 14.5.1.1.1 Normal Mode

In normal mode ( $\text{MODE} = 0$ ), not to be confused with SSI's Normal Mode, the port is connected in a single device point-to-single device port configuration (as a master or a slave) and the  $\text{RXDSEL}[2:0]$  setting selects the transmit signal from any port. In normal mode, any data format can be used (that is, SSI normal mode, SSI network mode, AC-97, and others).

If a user wishes to transmit SSI1's data (TXD) to port 5, PDCR5's  $\text{RXDSEL}[2:0]$  must be set to 000b. If the clock (TCK) and frame clock (TFS) are sent out to the external device from SSI1, PTCR5's  $\text{TFSEL}[3:0]$  and  $\text{TCSEL}[3:0]$  must be set to 0000b (port1) and 0000b (port1), while setting PTCR5's  $\text{TFSDIR}$  and  $\text{TCLKDIR}$  to 1.

If either or both of the clocks are to be received from external device to SSI1, PTCR5's TCLKDIR and/or TFSDIR must be set to 0 (input), and PTCR1's RFSEL[3:0] and/or TCLKDIR[3:0] to 0100b (port5).

Likewise, if a user wishes to receive serial data from Port 4 and send the receiving data to SSI2's RXD, one must set PDCR2's RXDSEL[2:0] to 11b. If the frame (RXFS) and bit clocks (RXC) are to be received from the external device, PTCR2's RFSEL[3:0] and RCSEL[3:0] must be set to 011b while PTCR4's RFSDIR and RCKDIR set to 1.

#### 14.5.1.1.2 Internal Network Mode

In internal network mode (MODE = 1), the output of the AND gate is routed (via the output of the port) to the RXD signal of the corresponding host interface.

The INMMASK bit vector selects the transmit signals of the ports that are to be connected in network mode. The transmit signals received at the AUDMUX ports (TxDn\_in) are ANDed together to form the output. In internal network mode, only one device can be transmitting in its predesignated timeslot and all other transmit signals must remain high (be in high-impedance state and pulled-up). Therefore, non-active signals in the selection will be high and do not influence the output of the AND gate.

Network mode is a protocol where a master SSI is connected to more than one slave SSI device and communication occurs on a time-slotted frame. Though network mode can allow master-slave and slave-slave communication, internal network mode supports only master-slave communication.

There are two scenarios where internal network mode can be used with external network mode:

1. Slave-only devices are attached to an external port.
2. A master device is attached to an external port and all slave devices connected to the same external port are disabled.

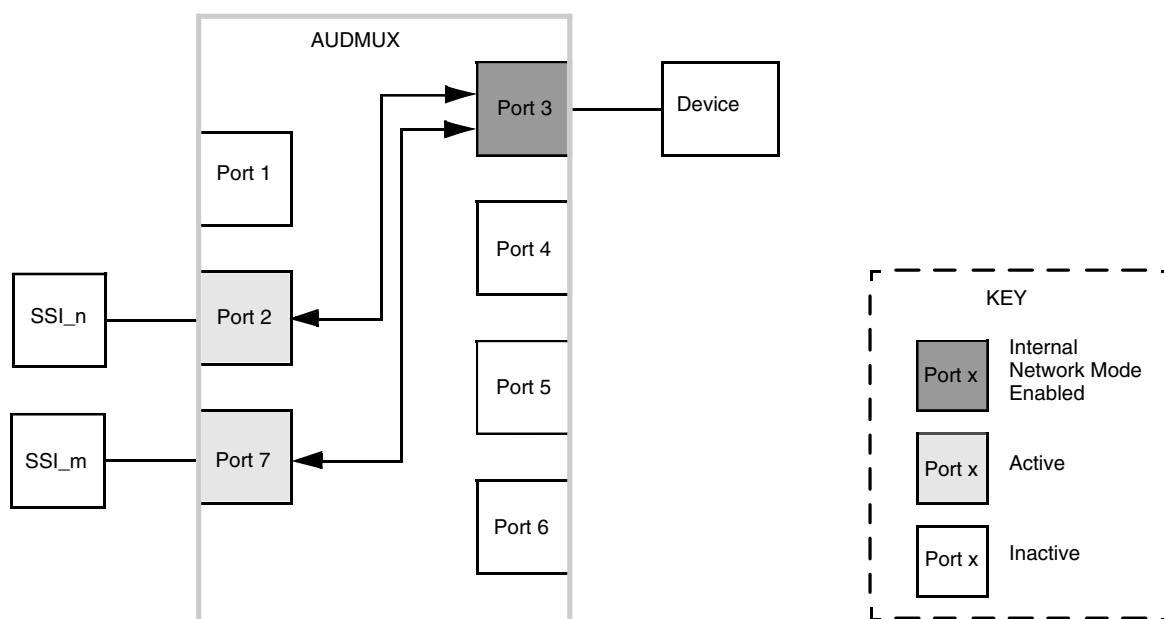
#### NOTE

When internal network mode is enabled at an external port, RXDSEL[3:0] for RxDn\_obe selection is ignored and RxD\_obe is always driven high (that is, asserted for all timeslots). All slave devices connected to the same port must be disabled.

#### Internal Network Mode Example 1

SSI\_m and SSI\_n are used with Port 3 in internal network mode as shown in [Figure 14-4](#). No pull-up resistors are required because the interfaces combined in internal network mode are on-chip interfaces.

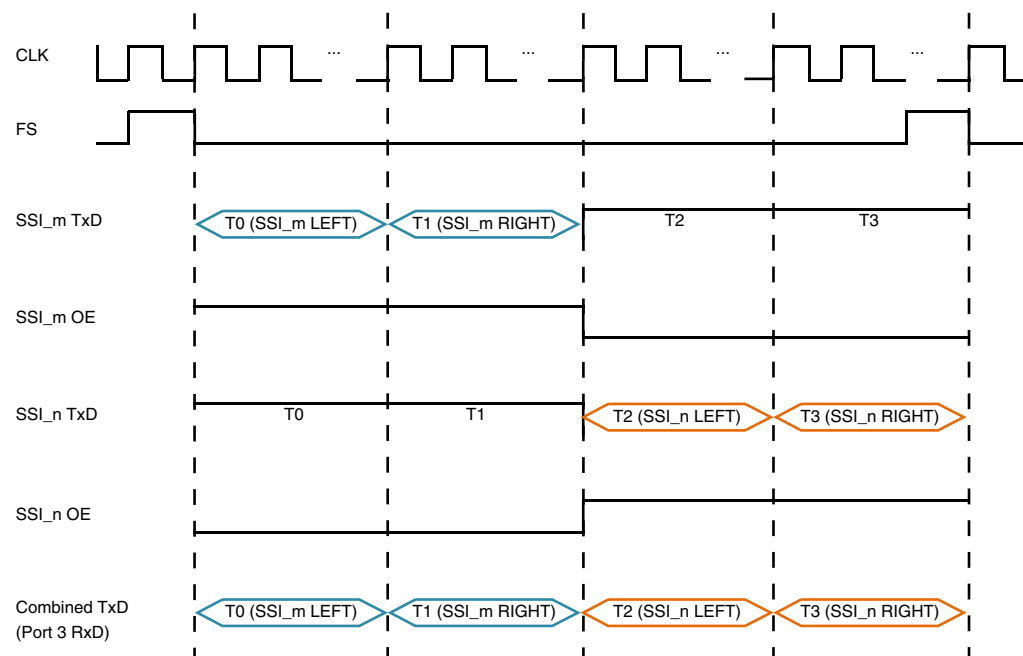




**Figure 14-4. Block Diagram For Example 1**

See [Figure 14-5](#) for the timing diagram of Example 1. The clock and frame sync signals show the bit and frame timing for the serial bus. The vertical dashed lines divide the frame into four timeslots.

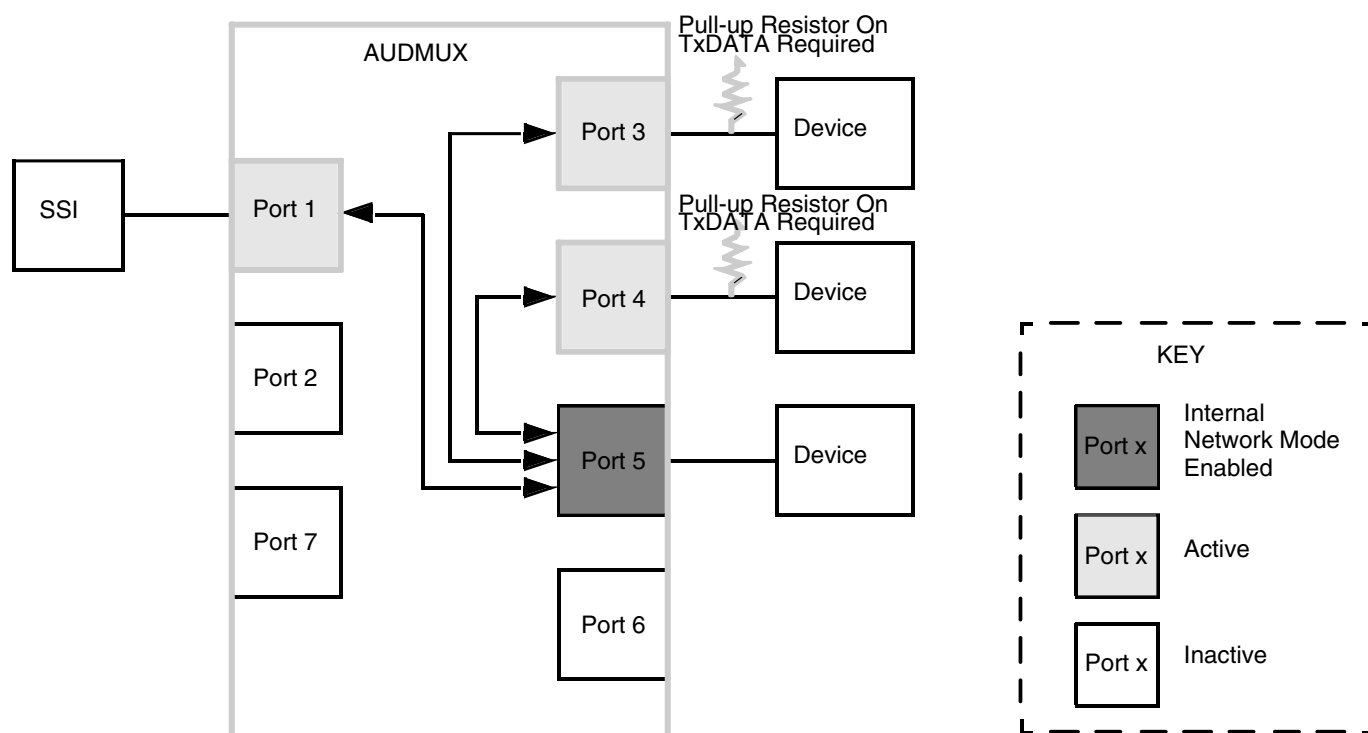
The data lines for SSI\_m and SSI\_n (as well as their output enables) are shown. Note that the on-chip interfaces drive a logic '1' when their output enables are logic '0'. The combined TXD line, which is the logical AND of the individual TXD lines, is used for Port 3's TXD line.



**Figure 14-5. Example Using Internal Ports For Transmit Data**

### Internal Network Mode Example 2

The SSI, Port 3, and Port 4 are used with Port 5 in internal network mode, as shown in the following figure. Note that Port 3 and Port 4 are external ports. Therefore, pull-up resistors are required on the Port 3 RXD and Port 4 RXD pins. This example shows the timing associated with using adjacent timeslots for the SSI, Port 3 and Port 4 .



**Figure 14-6. Block Diagram For Example 2**

The resistance value of the pull-up resistors must be sufficiently high such that a value of '0' can be pulled up to logic '1' within half of a period of the bitclock. The required resistance must be no larger than:

$R_{max} = 1 / (2 * f_{bc} * C)$  where:

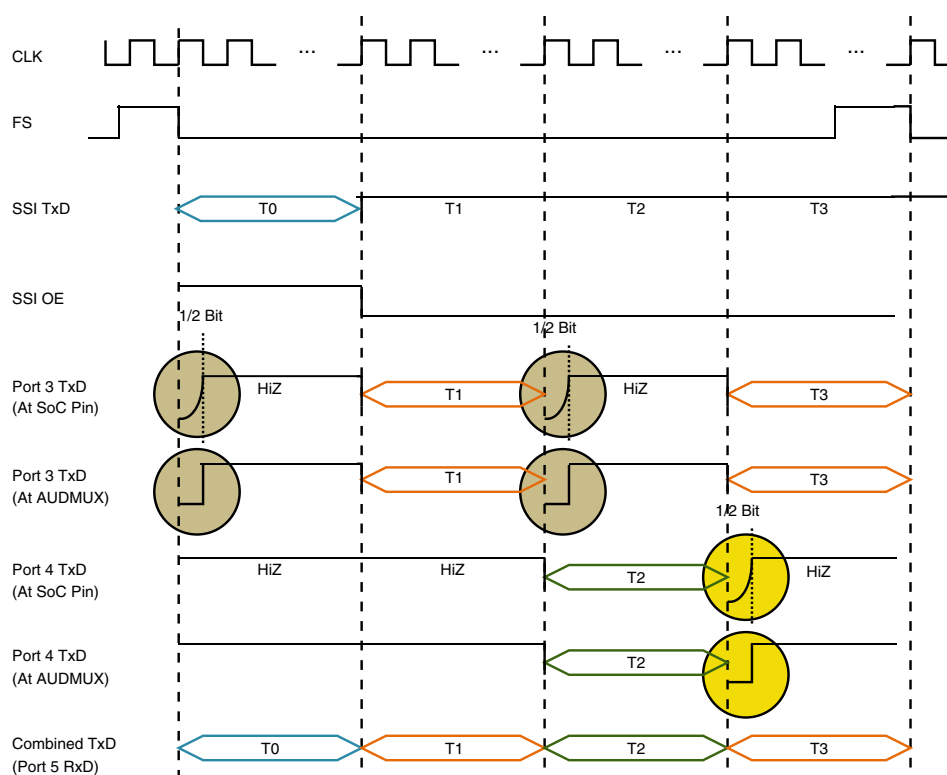
- $f_{bc}$  is the frequency of the bitclock
- $C$  is the total system capacitance (ICs, board traces, and so on)

The following figure shows the timing diagram for this example. The clock and frame sync signals show the bit and frame timing for the serial bus. The vertical dashed lines divide the frame into four timeslots.

The data lines for the SSI, Port 3 and Port 4 are shown. Note that the SSI transmits a logic '1' when its corresponding output enable is a logic '0'. The data lines from Port 3 and Port 4 at the pad are pulled high by pull-up resistors when they are in the high-impedance state. The data lines from Port 3 and Port 4 at the AUDMUX are pure digital signals and are constantly driven. The combined TXD line, which is the logical AND of the SSI, Port 3 and Port 4's TXD lines, is used for Port 5's TXD line.

Note the highlighted areas in the [Figure 14-7](#). This shows the transition time that occurs while a TXD line is being pulled high. In this example, this transition time is a maximum of 1/2 the period of the serial bitclock. This prevents corruption of the first data bit of the next timeslot. It is critical that the pull-up resistance is sufficient for the given bitclock frequency and system capacitance.

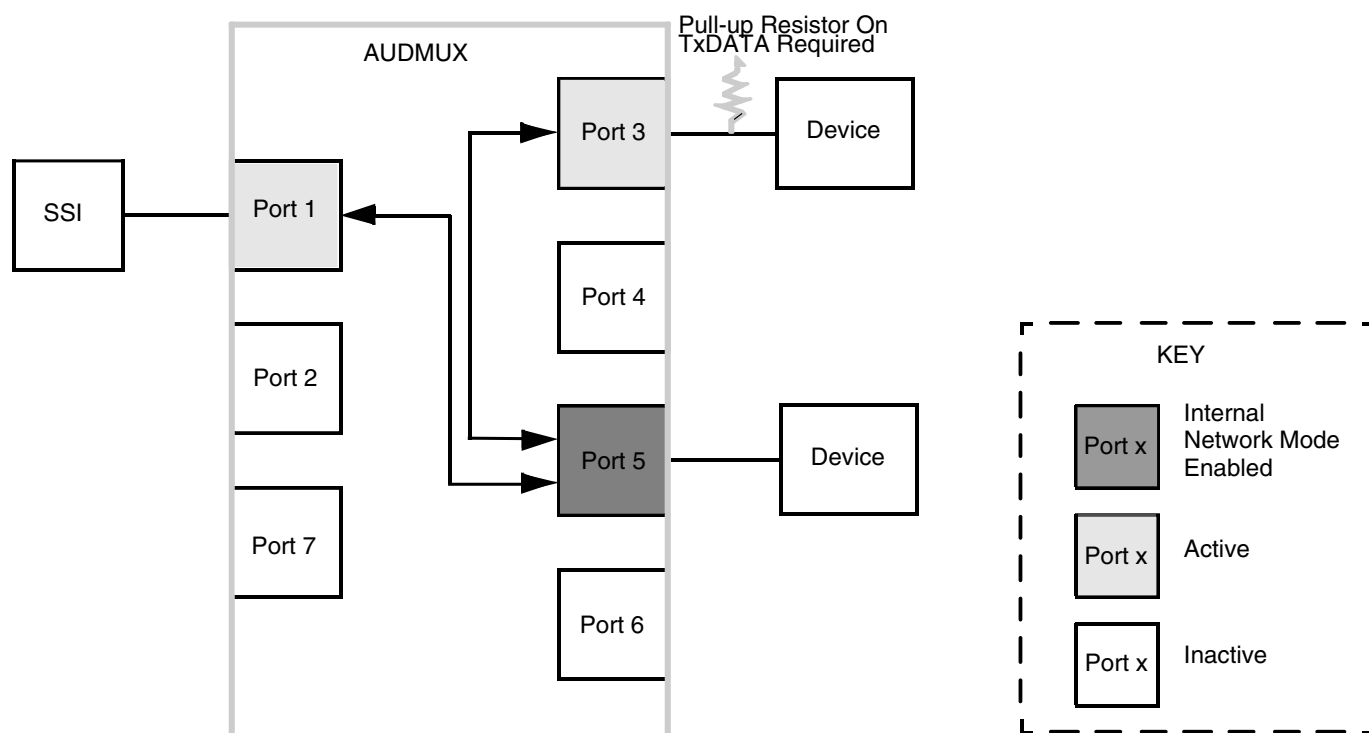
Note that hysteresis should be enabled at Port 3's RXD pad and Port 4's RXD pad to prevent the digital signals created by the pad from toggling rapidly during the pull-up period. The pads typically require a transition within 25ns unless hysteresis is enabled. Instead of using hysteresis, one could select a pull-up resistor sufficiently high to pull-up the signal at the pad within 25 ns; however, that would result in a higher resistance value and higher current drain.



**Figure 14-7. Example Using External Ports for Transmit Data in Consecutive Timeslots**

### Internal Network Mode Example 3

The SSI and Port 3 are used with Port 5 in internal network mode as shown in the following figure. Note that Port 3 is an external port. Therefore, a pull-up resistor is required on the Port 3TXD pin. This example shows the timing associated with inserting empty timeslots after the timeslots have been used by external ports.



**Figure 14-8. Block Diagram For Example 3**

The resistance value of the pull-up resistors must be sufficiently high such that a value of '0' can be pulled up to logic '1' by the time that the next occupied timeslot occurs. This allows a much weaker pull-up to be used as compared to Example 2. The required resistance must be no larger than:

$R_{max} = (4 * n + 1) / (2 * f_{bc} * C)$  where:

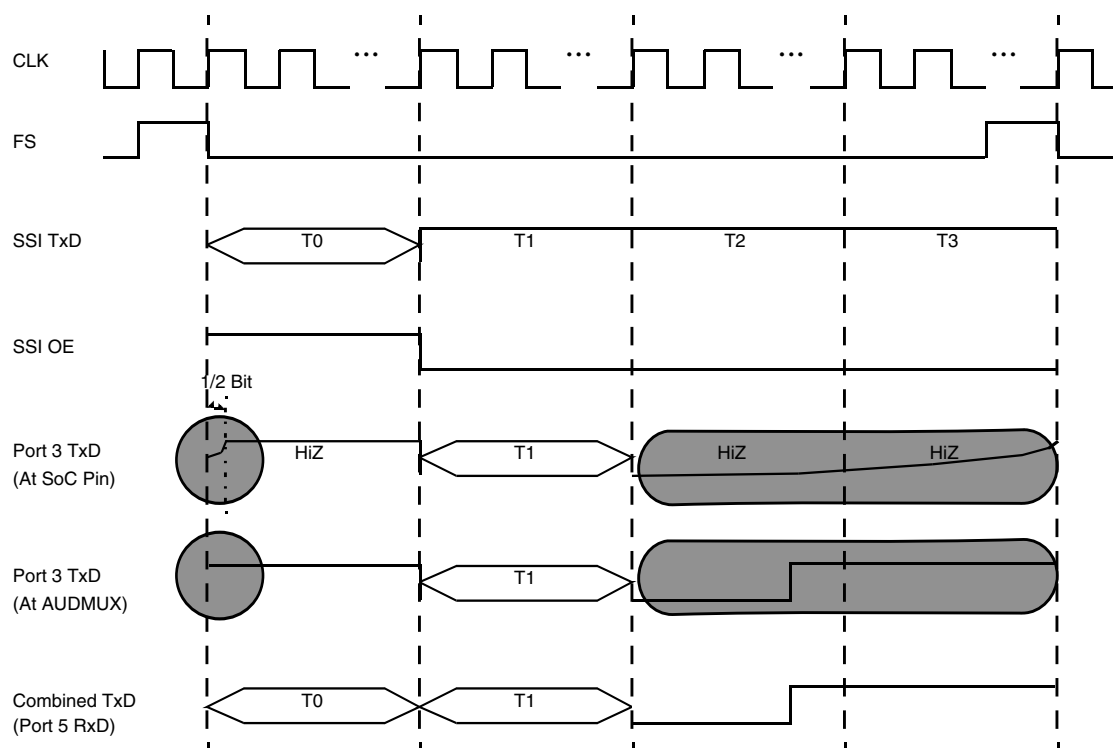
- $n$  is the number of bits per timeslot
- $f_{bc}$  is the frequency of the bitclock
- $C$  is the total system capacitance (ICs, board traces, and so on)

The figure below shows the timing diagram for this example. The clock and frame sync signals show the bit and frame timing for the serial bus. The vertical dashed lines divide the frame into four timeslots.

The data lines for the SSI and Port 3 are shown. Note that the SSI transmits a logic '1' when its corresponding output enable is a logic '0'. The data line from Port 3 at the pad is pulled high by a pull-up resistor when they are in the high-impedance state. The data line from Port 3 at the AUDMUX is a pure digital signal and is constantly driven. The combined TXD line, which is the logical AND of the SSI and Port 3's TXD lines, is used for Port 5's RXD line.

Note the highlighted area in the [Figure 14-9](#). This shows the transition time that occurs while Port 3's TXD line is being pulled high. In this example, this transition time is a maximum of two timeslots plus 1/2 the period of the serial bitclock. This prevents corruption of the first data bit of the next timeslot. It is critical that the pull-up resistance is sufficient for the given bitclock frequency and system capacitance.

Note that hysteresis must be enabled at Port 3's RXD pad to prevent the digital signal created by the pad from toggling rapidly during the extended pull-up period. The pads typically require a transition within 25 ns unless hysteresis is enabled.



**Figure 14-9. Example Using External Ports For Transmit Data In Nonconsecutive Timeslots**

#### 14.5.1.1.3 Transmit Data Output Enable Assertion

The TXD line from the internal network mode master (connected at any internal port) is put into the high-impedance state at the pad depending upon the assertion or deassertion of TxD\_obe, its corresponding output enable generated by the network mode master.

In the case of an external network mode master (connected at an external port), the corresponding TxD\_obe is always asserted after the port data register configuration.

### 14.5.1.2 Tx/Rx Switch and External Network Mode

External network mode is the traditional network mode connection. It is called external network mode to differentiate from the internal network mode. In external network mode, devices are connected to a single external port in a star or multi-drop configuration.

In network mode, there can be only one master (driving the frame sync and clock source) with the other devices configured in normal slave mode or network slave mode. Unlike internal network mode, both master-slave and slave-slave communication can take place in external network mode. Codec devices transmit on a single timeslot while processor serial interfaces (that is, SSI) can process more than one timeslot of data while in network master or slave mode.

The following figure shows the Tx/Rx data switch. `RxD_obe` is the output buffer enable signal and `RxD_out` is the data transmit signal from the serial interface. The `TxD_in` signal is the receive data signal going towards the `RXDSEL` muxes of all ports.

`D_TxRx` is the data pin which serves as the chip-level transmit data pin when the TxRx switch is not enabled. `D_RxTx` is the data pin which serves as the chip-level receive data pin when the TxRx switch is not enabled. The roles of these pins are reversed when the TxRx switch is enabled.

When `TXRXEN` is disabled (`TXRXEN=0`), `RxD_out` is routed to `D_TxRx` and `D_RxTx` is routed to `TxD_in`. The output buffer enable, selected by `RXDSEL[2:0]`, is routed to `Db_obe`.

When the Tx/Rx switch is enabled (`TXRXEN=1`), `RxD_out` is routed to `D_RxTx` and `D_TxRx` is routed to `TxD_in`. The output buffer enable, selected by `RXDSEL[2:0]`, is routed to `Da_obe`.

If the `RXDSELn[2:0]` field for any Port *n* is configured to select data from an internal port, the output buffer enable is selected by `RXDSELn[2:0]` and is routed to `Dan_obe/Dbn_obe`. In the case when the `RXDSELn[2:0]` field for Port *n* is configured to select data from an external port, the output buffer enable is always high and routed to `Dan_obe/Dbn_obe`, depending on the `TXRXENn` switch configuration.

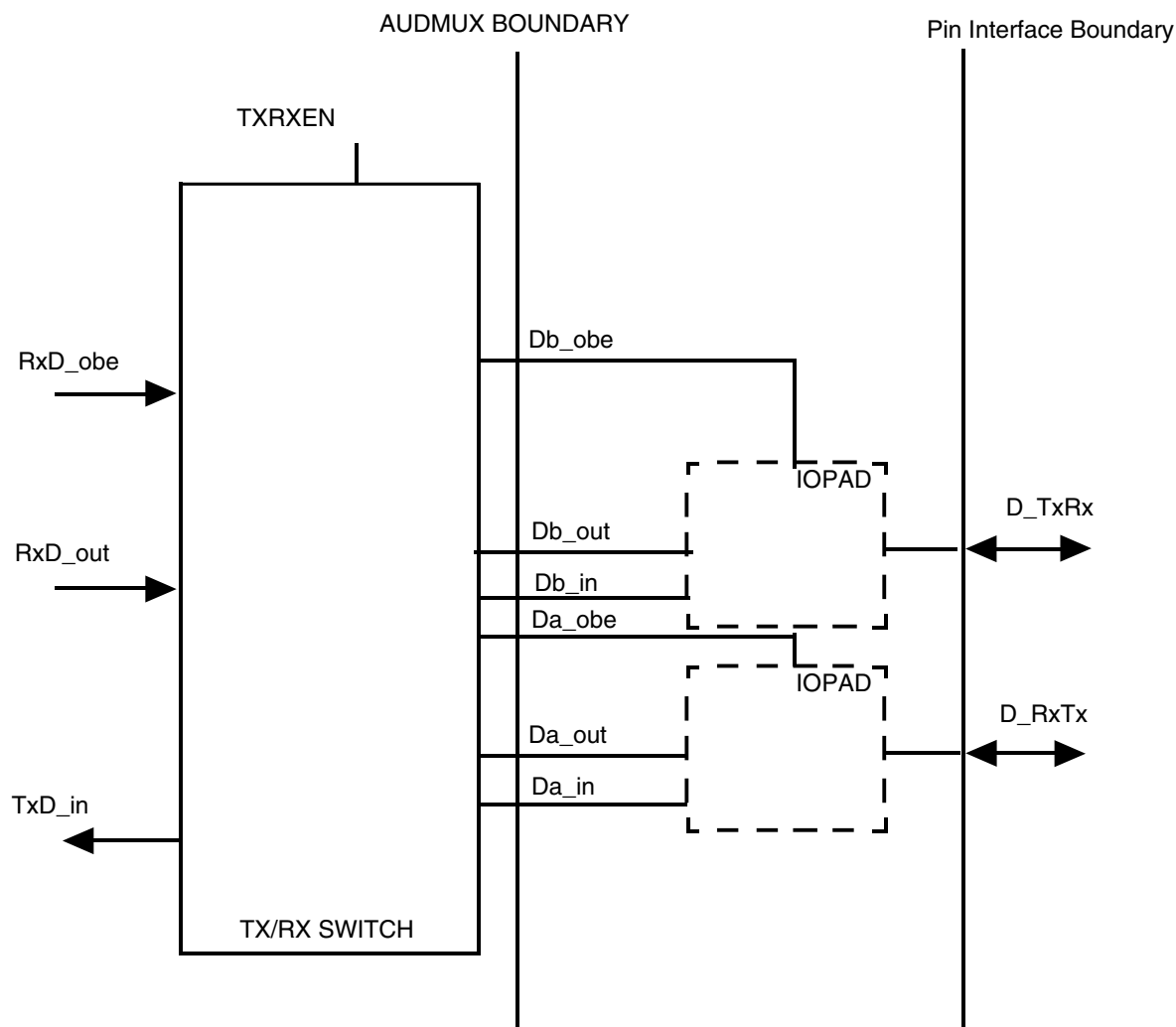


Figure 14-10. Tx/Rx Switch

### 14.5.1.3 Timing Modes

The AUDMUX ports are constructed as 6-wire interfaces. However, they can be used either in synchronous or asynchronous modes as determined by the SYN bit.

#### 14.5.1.3.1 Synchronous Mode (4-Wire Interface)

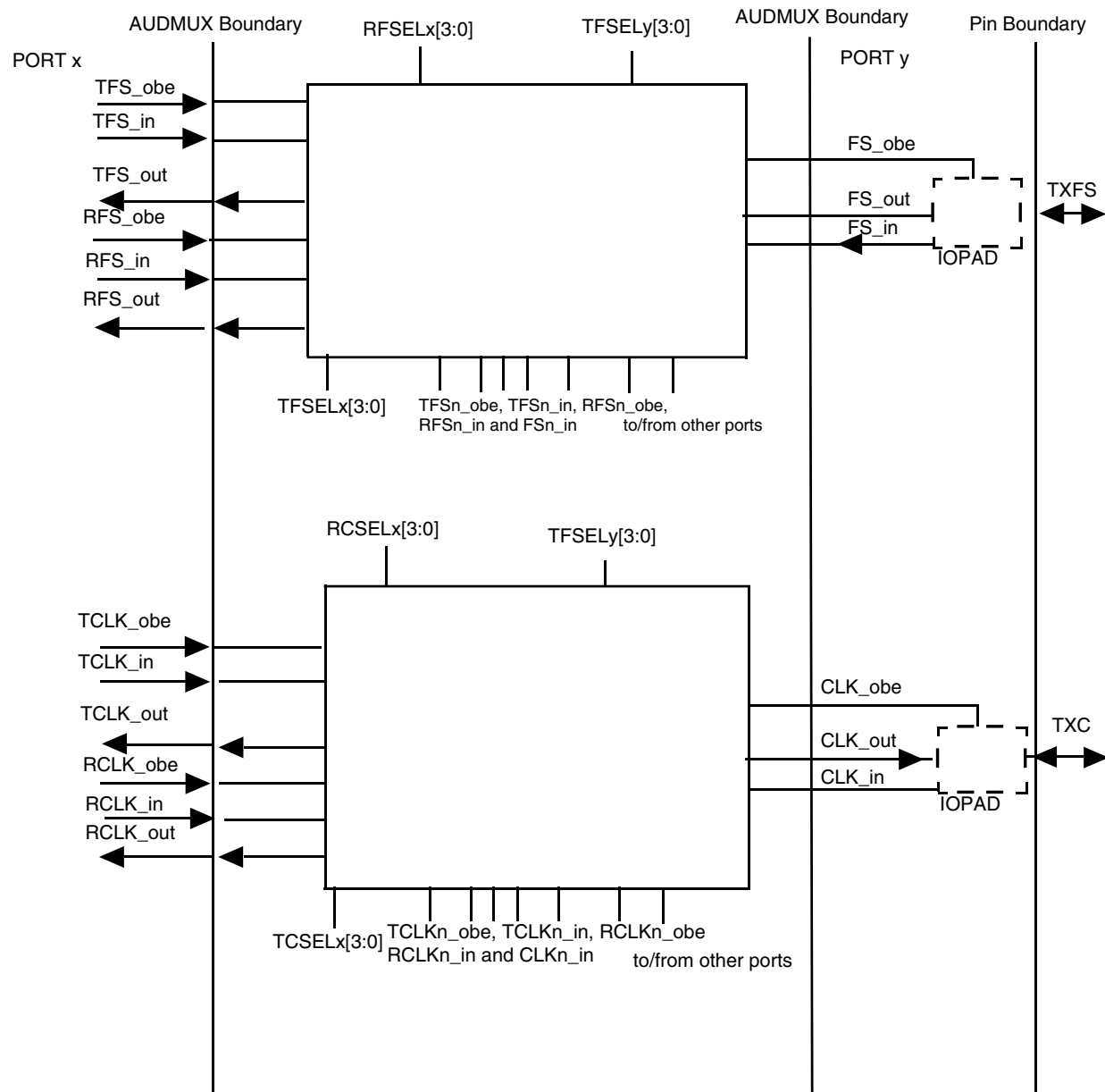
In Synchronous mode, each port set in SYN=1 will be a 4-wire interface (that is, RXD, TXD, TXC, TXFS). In this setup, receive data timing is determined by TXC and TXFS..



As shown in the following figure, SSI signals interfaced to Port x signals are routed to Port y. When  $SYN = 1$  the 6-wire signal from SSI is reduced to 4-wire within the internal logic.

TFS\_in, RFS\_in, TCLK\_in, and RCLK\_in are the input frame sync and bit clocks from the serial interface (Port x) with their corresponding output buffer enable signals (\_obe). TFS\_out, RFS\_out, TCLK\_out, and RCLK\_out are the frame sync and bit clocks that are transmitted to the serial interface from the other ports.

The TFS\_out and TCLK\_out are selected at Port x by the TFSEL and TCSEL mux settings, respectively. RFS\_out and RCLK\_out are selected at Port x by the RFSEL and RCSEL mux settings, respectively. Similarly, in the external direction, Port y is configured as a 4-wire port; TFSEL selects the FS\_obe and FS\_out signals. In this mode, the configuration of RFSEL and RCSEL is not used, since the RFS\_out and RCLK\_out pins at Port y are not available.



**Figure 14-11. Frame Sync and Clock Routing When External Port Is 4-Wire**

#### 14.5.1.3.2 Asynchronous Mode (6-Wire Interface)

In Asynchronous mode, the port has a 6-wire interface (meaning RXD, TXD, TXC, TXFS, RXC, RXFS). This mode has additional receive clock (RXC) and frame sync (RXFS) signals as compared to the synchronous or 4-wire interface.

As shown in the figures below, Port x signals can be routed to Port y, producing 6-wire to 6-wire port connectivity.

TFS\_in, RFS\_in, TCLK\_in, and RCLK\_in are input frame sync and bit clocks from the serial interface (Port x) with their corresponding output buffer enable signals (\_obe). TFS\_out, RFS\_out, TCLK\_out, and RCLK\_out are the frame sync and bit clocks that are transmitted to the serial interface from the other ports.

TFS\_out and TCLK\_out are selected by the TFSEL and TCSEL mux settings, respectively. RFS\_out and RCLK\_out are selected by the RFSEL and RCSEL mux settings, respectively. Similarly, in the external direction, the TFSEL selects the TxFS\_obe and TxFS\_out signals and TCSEL selects the TxCLK\_obe and TxClk\_out signals. The RFSEL selects the RxFS\_obe and RxFS\_out signals and RCSEL selects the RxCLK\_obe and RxCLK\_out signals.

### NOTE

Because FS\_in and CLK\_in from external interfaces are also routed to the TFSEL and TCSEL muxes of the external ports, these signals do not have corresponding buffer enable signals. Consequently, their corresponding inputs to the TFSEL and TCSEL mux of the external ports have to be tied high.

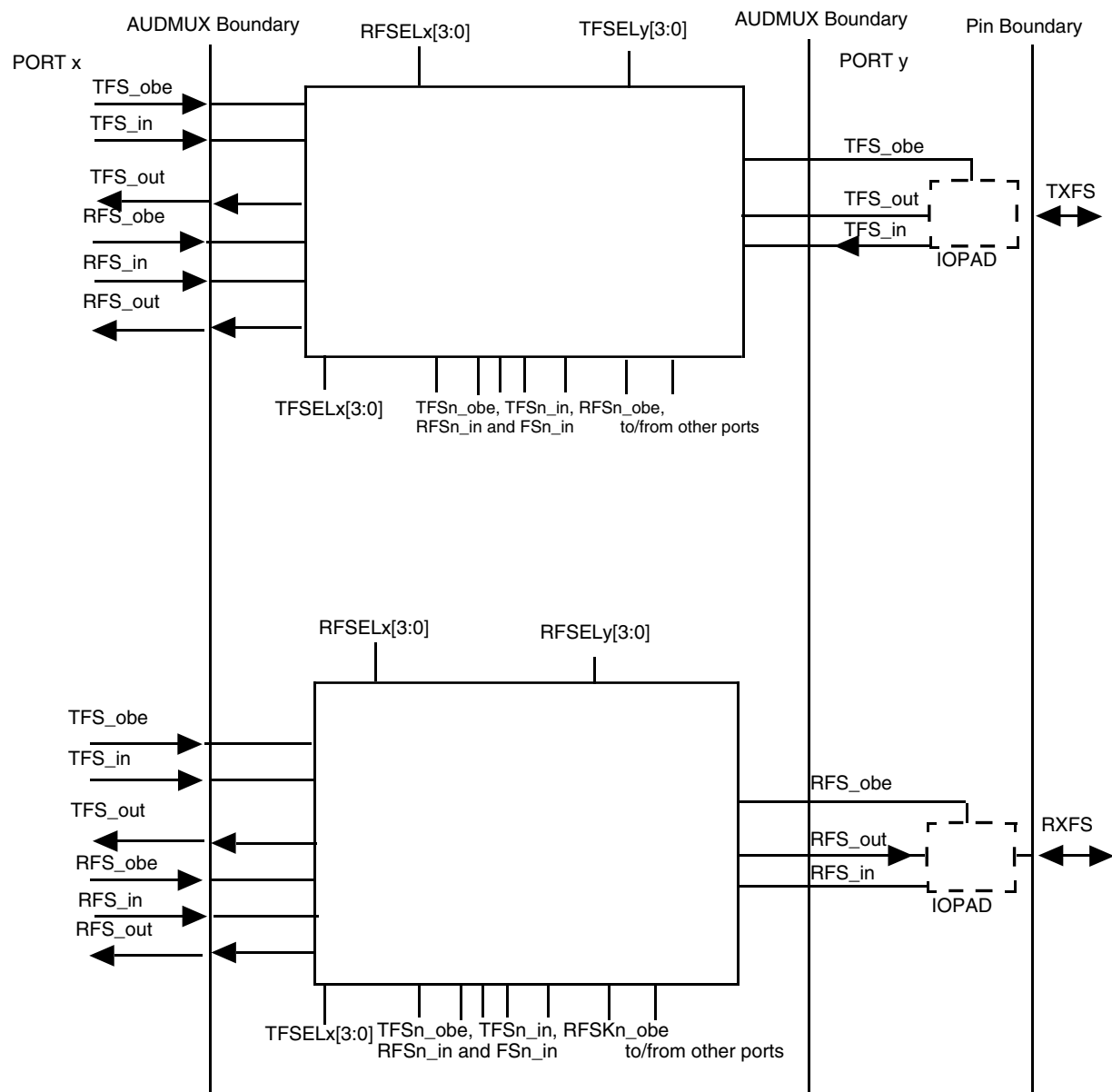


Figure 14-12. Frame Sync Routing When External Port Is 6-Wire



**Figure 14-13. Clock Routing When External Port Is 6 Wire**

### 14.5.2 Connectivity Between Ports

Four basic types of connections are provided by the AUDMUX:

- Internal port to external port
- External port to external port
- Internal port to internal port
- Loopback

The corresponding data connections are described in the following sections.

### 14.5.2.1 Internal Port to External Port Connectivity

The internal port is connected to a processor's serial interface. TxD\_obe is the buffer enable signal from the serial interface, TxD\_in is the input transmit data from the serial interface to the AUDMUX, and RxD\_out is the receive data output from the AUDMUX to the serial interface.

RXDSEL[2:0] of the external port selects the buffer enable signal (TxD\_obe) and transmit data output (TxD\_out) signal from the TxD\_obe and RxD\_in signals. RXDSEL[2:0] is a common signal to both selection muxes.

#### NOTE

Because buffer TxD\_in signals from external interfaces do not have corresponding buffer enable signals, their buffer enable signals into the selection mux are tied high. This will ensure that selection of TxD\_in, as RxD\_out will also drive the RxD\_obe output high.

Transmit Data from the serial interface goes into the RXDSEL data mux and comes out as RxD\_out. RxD\_out is routed to Da\_TxRx when TXRXEN is disabled and to D\_RxTx when TXRXEN is enabled. Similarly, D\_RxTx is routed to TxD\_in when TXRXEN is disabled and D\_TxRx is routed to TxD\_in when TXRXEN is enabled. The routing of frame syncs is shown in [Figure 14-12](#) and the routing of interface clocks is shown in [Figure 14-10](#).

If internal network mode is disabled, then RXDSEL selects the TxD\_in, which is sent from the AUDMUX to the serial interface connected at Port x. When the internal network mode is selected, RxD\_out is constructed by ANDing selected TxD\_in signals from the ports (as determined by INMMASK).

If there is more than one device attached to the external port at D\_TxRx and D\_RxTx and one of the devices is a network master, then two conditions must be noted:

1. When the external master is enabled in network mode, then the serial interface at Port x must be configured as a slave (normal or network mode). No Tx/Rx switching is required.
2. When the external master is disabled and the serial interface at Port x and other slave devices must communicate, then the serial interface at Port x must be configured as a network mode master and the Tx/Rx switch at Port y must be enabled (TXRXEN=1). This will ensure that the transmit and receive paths are connected appropriately.

To communicate with more than one port, internal network mode can be enabled at Port x. In internal network mode, it is possible to communicate with any device attached to the other ports. Internal network mode shall be enabled at the port that is the SSI network mode master.

### 14.5.2.2 External Port to External Port Connectivity

External ports can communicate with external ports directly.

External ports can communicate together in three ways:

1. Each port's receive logic is configured in normal mode (MODE = 0) . Each port's RXDSEL[2:0] field is configured to select the other port's transmit data. Bit fields associated with clock/frame sync selection and direction are configured for each port. Either port can be the master.
2. One port is configured in internal network mode (MODE = 1) . All desired data lines are combined by the AND gate as determined by INMMASK[7:0]. Since an external port is being used as the internal network mode master, all other devices on the same AUDMUX port as the internal network mode master must be disabled. This configuration can be used with a combination of internal and external ports. All external ports must have a pull-up resistor on its RXD pin. Bit fields associated with clock/frame sync selection and direction are configured for each port. Any port can be the master.

### 14.5.2.3 Internal Port to Internal Port Connectivity

Internal ports can communicate with other internal ports directly, thereby providing a means for synchronous interprocessor communication.

Internal ports can communicate together in two ways:

1. Each port's receive logic is configured in normal mode (MODE = 0) . Each port's RXDSEL[2:0] field is configured to select the other port's transmit data. Bit fields associated with clock/frame sync selection and direction are configured for each port. Either port can be the master.
2. One port is configured in internal network mode (MODE = 1) . All desired data lines are combined by the AND gate as determined by INMMASK[7:0]. This configuration can be used with a combination of internal and external ports. All external ports must have a pull-up resistor on its RXD pin. Bit fields associated with clock/frame sync selection and direction are configured for each port. Any port can be the master.

### 14.5.2.4 Loopback Connectivity

AUDMUX ports can communicate with themselves in order to provide loopback functionality. Port x can route its TXD signal to its own RxD\_out signal by setting RXDSELx[2:0] to its own port number. This is supported by all ports in the AUDMUX.

In addition, ports can provide loopback support in internal network mode. With internal network mode, the internal network mode master can loop its TXD signal (combined with those of other ports, if desired) back into its RxD\_out signal. Port x's INMMASK should be set such that bit (x - 1) is clear in order to enable the loopback.

## 14.6 AUDMUX Memory Map/Register Definition

This section includes the block memory map and detailed descriptions of all registers. For the base address of a specific sub-block instantiation, see the system memory map in this manual.

The AUDMUX memory map is shown in the following table.

**AUDMUX memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
21D_8000	Port Timing Control Register 1 (AUDMUX_PTCR1)	32	R/W	AD40_0800h	<a href="#">14.6.1/469</a>
21D_8004	Port Data Control Register 1 (AUDMUX_PDCR1)	32	R/W	0000_A000h	<a href="#">14.6.2/471</a>
21D_8008	Port Timing Control Register 2 (AUDMUX_PTCR2)	32	R/W	A500_0800h	<a href="#">14.6.3/472</a>
21D_800C	Port Data Control Register 2 (AUDMUX_PDCR2)	32	R/W	0000_8000h	<a href="#">14.6.4/474</a>
21D_8010	Port Timing Control Register 3 (AUDMUX_PTCR3)	32	R/W	9CC0_0800h	<a href="#">14.6.5/475</a>
21D_8014	Port Data Control Register 3 (AUDMUX_PDCR3)	32	R/W	0000_6000h	<a href="#">14.6.6/477</a>
21D_8018	Port Timing Control Register 4 (AUDMUX_PTCR4)	32	R/W	0000_0800h	<a href="#">14.6.7/478</a>
21D_801C	Port Data Control Register 4 (AUDMUX_PDCR4)	32	R/W	0000_4000h	<a href="#">14.6.8/480</a>
21D_8020	Port Timing Control Register 5 (AUDMUX_PTCR5)	32	R/W	0000_0800h	<a href="#">14.6.9/481</a>
21D_8024	Port Data Control Register 5 (AUDMUX_PDCR5)	32	R/W	0000_2000h	<a href="#">14.6.10/483</a>
21D_8028	Port Timing Control Register 6 (AUDMUX_PTCR6)	32	R/W	0000_0800h	<a href="#">14.6.11/484</a>
21D_802C	Port Data Control Register 6 (AUDMUX_PDCR6)	32	R/W	0000_0000h	<a href="#">14.6.12/486</a>
21D_8030	Port Timing Control Register 7 (AUDMUX_PTCR7)	32	R/W	0000_0800h	<a href="#">14.6.13/487</a>
21D_8034	Port Data Control Register 7 (AUDMUX_PDCR7)	32	R/W	0000_C000h	<a href="#">14.6.14/489</a>



## 14.6.1 Port Timing Control Register 1 (AUDMUX\_PTCR1)

PTCR1 is the Port Timing Control Register for Port 1.

Address: 21D\_8000h base + 0h offset = 21D\_8000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	TFS_DIR		TFSEL[3:0]				TCLKDIR	TCSEL[3:0]				RFS_DIR	RFSEL[3:0]				RCLKDIR
W																	
Reset	1	0	1	0	1	1	0	1	0	1	0	0	0	0	0	0	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	RCSEL[3:0]				SYN	0											
W																	
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	

### AUDMUX\_PTCR1 field descriptions

Field	Description
31 TFS_DIR	Transmit Frame Sync Direction Control. This bit sets the direction of the TXFS pin of the interface as an output or input. When set as an input, the TFSEL settings are ignored. When set as an output, the TFSEL settings determine the source port of the frame sync.  0 TXFS is an input. 1 TXFS is an output.
30–27 TFSEL[3:0]	Transmit Frame Sync Select. Selects the source port from which TXFS is sourced.  0xxx Selects TXFS from port. 1xxx Selects RXFS from port. x000 Port 1 x110 Port 7 x111 Reserved
26 TCLKDIR	Transmit Clock Direction Control. This bit sets the direction of the TXC pin of the interface as an output or input. When set as an input, the TCSEL settings are ignored. When set as an output, the TCSEL settings determine the source port of the clock.  0 TXC is an input. 1 TXC is an output.
25–22 TCSEL[3:0]	Transmit Clock Select. Selects the source port from which TXC is sourced.  0xxx Selects TXC from port. 1xxx Selects RXC from port. x000 Port 1

Table continues on the next page...

## AUDMUX\_PTCR1 field descriptions (continued)

Field	Description
	x110 Port 7 x111 Reserved
21 RFS_DIR	Receive Frame Sync Direction Control. This bit sets the direction of the RXFS pin of the interface as an output or input. When set as an input, the RFSEL settings are ignored. When set as an output, the RFSEL settings determine the source port of the frame sync.  0 RXFS is an input. 1 RXFS is an output.
20–17 RFSEL[3:0]	Receive Frame Sync Select. Selects the source port from which RXFS is sourced. RXFS can be sourced from TXFS and RXFS from other ports.  0xxx Selects TXFS from port. 1xxx Selects RXFS from port. x000 Port 1 x110 Port 7 x111 Reserved
16 RCLKDIR	Receive Clock Direction Control. This bit sets the direction of the RXC pin of the interface as an output or input. When set as an input, the RCSEL settings are ignored. When set as an output, the RCSEL settings determine the source port of the clock.  <b>NOTE:</b> RCLKDIR and SYN should not be changed at the same time.  0 RXC is an input. 1 RXC is an output.
15–12 RCSEL[3:0]	Receive Clock Select. Selects the source port from which RXC is sourced. RXC can be sourced from TXC and RXC from other ports.  0xxx Selects TXC from port. 1xxx Selects RXC from port. x000 Port 1 x110 Port 7 x111 Reserved
11 SYN	Synchronous/Asynchronous Select. When SYN is set, synchronous mode is chosen and the transmit and receive sections use common clock and frame sync signals (that is, the port is a 4-wire interface). When SYN is cleared, asynchronous mode is chosen and separate clock and frame sync signals are used for the transmit and receive sections (that is, the port is a 6-wire interface).  <b>NOTE:</b> RCLKDIR and SYN should not be changed at the same time.  0 Asynchronous mode 1 Synchronous mode (default)
10–0 Reserved	This read-only field is reserved and always has the value 0.

## 14.6.2 Port Data Control Register 1 (AUDMUX\_PDCR1)

PDCR1 is the Port Data Control Register for Port 1.

Address: 21D\_8000h base + 4h offset = 21D\_8004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDSEL[2:0]			TXRXEN	0			MODE	INMMASK[7:0]							
W																
Reset	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

**AUDMUX\_PDCR1 field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–13 RXDSEL[2:0]	Receive Data Select. Selects the source port for the RXD data. RXDSEL is ignored if MODE = 1 (that is, Internal Network Mode is enabled).  xxx    Port number for RXD 000    Port 1 110    Port 7 111    Reserved
12 TXRXEN	Transmit/Receive Switch Enable. Swaps the transmit and receive signals.  0    No switch (Transmit Pin = Transmit, Receive Pin = Receive) 1    Switch (Transmit Pin = Receive, Receive Pin = Transmit)
11–9 Reserved	This read-only field is reserved and always has the value 0.
8 MODE	Mode Select. This field selects the mode in which the port is to operate. The modes of operation include the following: <ul style="list-style-type: none"> <li>Normal mode, in which the RXD from the port selected by RXDSEL is routed to the port.</li> <li>Internal Network mode in which RXD from other ports are ANDed together. RXDSEL is ignored. INMMASK determines which RXD signals are ANDed together.</li> </ul> 0    Normal mode 1    Internal Network mode

*Table continues on the next page...*

**AUDMUX\_PDCR1 field descriptions (continued)**

Field	Description
7-0 INMMASK[7:0]	Internal Network Mode Mask. Bit mask that selects the ports from which the RXD signals are to be ANDed together for internal network mode. Bit 6 represents RXD from Port 7 and bit0 represents RXD from Port 1.  0 Includes RXDn for ANDing 1 Excludes RXDn from ANDing

**14.6.3 Port Timing Control Register 2 (AUDMUX\_PTCR2)**

PTCR2 is the Port Timing Control Register for Port 2.

Address: 21D\_8000h base + 8h offset = 21D\_8008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	1	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0
W	1	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0
Reset	1	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	1	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0
W	1	0	1	0	0	1	0	1	0	0	0	0	0	0	0	0
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

**AUDMUX\_PTCR2 field descriptions**

Field	Description
31 TFS_DIR	Transmit Frame Sync Direction Control. This bit sets the direction of the TXFS pin of the interface as an output or input. When set as an input, the TFSEL settings are ignored. When set as an output, the TFSEL settings determine the source port of the frame sync.  0 TXFS is an input. 1 TXFS is an output.
30-27 TFSEL[3:0]	Transmit Frame Sync Select. Selects the source port from which TXFS is sourced.  0xxx Selects TXFS from port. 1xxx Selects RXFS from port. x000 Port 1 x110 Port 7 x111 Reserved

Table continues on the next page...

**AUDMUX\_PTCR2 field descriptions (continued)**

Field	Description
26 TCLKDIR	Transmit Clock Direction Control. This bit sets the direction of the TXC pin of the interface as an output or input. When set as an input, the TCSEL settings are ignored. When set as an output, the TCSEL settings determine the source port of the clock.  0 TXC is an input. 1 TXC is an output.
25–22 TCSEL[3:0]	Transmit Clock Select. Selects the source port from which TXC is sourced.  0xxx Selects TXC from port. 1xxx Selects RXC from port. x000 Port 1 x110 Port 7 x111 Reserved
21 RFS_DIR	Receive Frame Sync Direction Control. This bit sets the direction of the RXFS pin of the interface as an output or input. When set as an input, the RFSEL settings are ignored. When set as an output, the RFSEL settings determine the source port of the frame sync.  0 RXFS is an input. 1 RXFS is an output.
20–17 RFSEL[3:0]	Receive Frame Sync Select. Selects the source port from which RXFS is sourced. RXFS can be sourced from TXFS and RXFS from other ports.  0xxx Selects TXFS from port. 1xxx Selects RXFS from port. x000 Port 1 x110 Port 7 x111 Reserved
16 RCLKDIR	Receive Clock Direction Control. This bit sets the direction of the RXC pin of the interface as an output or input. When set as an input, the RCSEL settings are ignored. When set as an output, the RCSEL settings determine the source port of the clock.  <b>NOTE:</b> RCLKDIR and SYN should not be changed at the same time.  0 RXC is an input. 1 RXC is an output.
15–12 RCSEL[3:0]	Receive Clock Select. Selects the source port from which RXC is sourced. RXC can be sourced from TXC and RXC from other ports.  0xxx Selects TXC from port. 1xxx Selects RXC from port. x000 Port 1 x110 Port 7 x111 Reserved
11 SYN	Synchronous/Asynchronous Select. When SYN is set, synchronous mode is chosen and the transmit and receive sections use common clock and frame sync signals (that is, the port is a 4-wire interface). When SYN is cleared, asynchronous mode is chosen and separate clock and frame sync signals are used for the transmit and receive sections (that is, the port is a 6-wire interface).  <b>NOTE:</b> RCLKDIR and SYN should not be changed at the same time.

*Table continues on the next page...*

**AUDMUX\_PTCR2 field descriptions (continued)**

Field	Description
0	Asynchronous mode
1	Synchronous mode (default)
10–0 Reserved	This read-only field is reserved and always has the value 0.

**14.6.4 Port Data Control Register 2 (AUDMUX\_PDCR2)**

PDCR2 is the Port Data Control Register for Port 2.

Address: 21D\_8000h base + Ch offset = 21D\_800Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDSEL[2:0]			TXRXEN	0			MODE	INMMASK[7:0]							
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**AUDMUX\_PDCR2 field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–13 RXDSEL[2:0]	Receive Data Select. Selects the source port for the RXD data. RXDSEL is ignored if MODE = 1 (that is, Internal Network Mode is enabled).  xxx Port number for RXD 000 Port 1 110 Port 7 111 Reserved
12 TXRXEN	Transmit/Receive Switch Enable. Swaps the transmit and receive signals.  0 No switch (Transmit Pin = Transmit, Receive Pin = Receive) 1 Switch (Transmit Pin = Receive, Receive Pin = Transmit)
11–9 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**AUDMUX\_PDCR2 field descriptions (continued)**

Field	Description
8 MODE	Mode Select. This field selects the mode in which the port is to operate. The modes of operation include the following: <ul style="list-style-type: none"> <li>Normal mode, in which the RXD from the port selected by RXDSEL is routed to the port.</li> <li>Internal Network mode in which RXD from other ports are ANDed together. RXDSEL is ignored. INMMASK determines which RXD signals are ANDed together.</li> </ul> 0 Normal mode 1 Internal Network mode
7–0 INMMASK[7:0]	Internal Network Mode Mask. Bit mask that selects the ports from which the RXD signals are to be ANDed together for internal network mode. Bit 6 represents RXD from Port 7 and bit0 represents RXD from Port 1. 0 Includes RXDn for ANDing 1 Excludes RXDn from ANDing

**14.6.5 Port Timing Control Register 3 (AUDMUX\_PTCR3)**

PTCR3 is the Port Timing Control Register for Port 3.

Address: 21D\_8000h base + 10h offset = 21D\_8010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TFS_DIR	TFSEL[3:0]				TCLKDIR	TCSEL[3:0]				RFS_DIR	RFSEL[3:0]				RCLKDIR
W																
Reset	1	0	0	1	1	1	0	0	1	1	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RCSEL[3:0]				SYN	0										
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

**AUDMUX\_PTCR3 field descriptions**

Field	Description
31 TFS_DIR	Transmit Frame Sync Direction Control. This bit sets the direction of the TXFS pin of the interface as an output or input. When set as an input, the TFSEL settings are ignored. When set as an output, the TFSEL settings determine the source port of the frame sync. 0 TXFS is an input. 1 TXFS is an output.

*Table continues on the next page...*

**AUDMUX\_PTCCR3 field descriptions (continued)**

Field	Description
30–27 TFSEL[3:0]	Transmit Frame Sync Select. Selects the source port from which TXFS is sourced.  0xxx Selects TXFS from port. 1xxx Selects RXFS from port. x000 Port 1 x110 Port 7 x111 Reserved
26 TCLKDIR	Transmit Clock Direction Control. This bit sets the direction of the TXC pin of the interface as an output or input. When set as an input, the TCSEL settings are ignored. When set as an output, the TCSEL settings determine the source port of the clock.  0 TXC is an input. 1 TXC is an output.
25–22 TCSEL[3:0]	Transmit Clock Select. Selects the source port from which TXC is sourced.  0xxx Selects TXC from port. 1xxx Selects RXC from port. x000 Port 1 x110 Port 7 x111 Reserved
21 RFS_DIR	Receive Frame Sync Direction Control. This bit sets the direction of the RXFS pin of the interface as an output or input. When set as an input, the RFSEL settings are ignored. When set as an output, the RFSEL settings determine the source port of the frame sync.  0 RXFS is an input. 1 RXFS is an output.
20–17 RFSEL[3:0]	Receive Frame Sync Select. Selects the source port from which RXFS is sourced. RXFS can be sourced from TXFS and RXFS from other ports.  0xxx Selects TXFS from port. 1xxx Selects RXFS from port. x000 Port 1 x110 Port 7 x111 Reserved
16 RCLKDIR	Receive Clock Direction Control. This bit sets the direction of the RXC pin of the interface as an output or input. When set as an input, the RCSEL settings are ignored. When set as an output, the RCSEL settings determine the source port of the clock.  <b>NOTE:</b> RCLKDIR and SYN should not be changed at the same time.  0 RXC is an input. 1 RXC is an output.
15–12 RCSEL[3:0]	Receive Clock Select. Selects the source port from which RXC is sourced. RXC can be sourced from TXC and RXC from other ports.  0xxx Selects TXC from port. 1xxx Selects RXC from port. x000 Port 1 x110 Port 7 x111 Reserved

*Table continues on the next page...*



**AUDMUX\_PTCR3 field descriptions (continued)**

Field	Description
11 SYN	Synchronous/Asynchronous Select. When SYN is set, synchronous mode is chosen and the transmit and receive sections use common clock and frame sync signals (that is, the port is a 4-wire interface). When SYN is cleared, asynchronous mode is chosen and separate clock and frame sync signals are used for the transmit and receive sections (that is, the port is a 6-wire interface).  <b>NOTE:</b> RCLKDIR and SYN should not be changed at the same time.  0 Asynchronous mode 1 Synchronous mode (default)
10–0 Reserved	This read-only field is reserved and always has the value 0.

**14.6.6 Port Data Control Register 3 (AUDMUX\_PDCR3)**

PDCR3 is the Port Data Control Register for Port 3.

Address: 21D\_8000h base + 14h offset = 21D\_8014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDSEL[2:0]			TXRXEN	0			MODE	INMMASK[7:0]							
W																
Reset	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

**AUDMUX\_PDCR3 field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–13 RXDSEL[2:0]	Receive Data Select. Selects the source port for the RXD data. RXDSEL is ignored if MODE = 1 (that is, Internal Network Mode is enabled).  xxx Port number for RXD 000 Port 1 110 Port 7 111 Reserved

Table continues on the next page...

**AUDMUX\_PDCR3 field descriptions (continued)**

Field	Description
12 TXRXEN	Transmit/Receive Switch Enable. Swaps the transmit and receive signals.  0 No switch (Transmit Pin = Transmit, Receive Pin = Receive) 1 Switch (Transmit Pin = Receive, Receive Pin = Transmit)
11–9 Reserved	This read-only field is reserved and always has the value 0.
8 MODE	Mode Select. This field selects the mode in which the port is to operate. The modes of operation include the following: <ul style="list-style-type: none"> <li>Normal mode, in which the RXD from the port selected by RXDSEL is routed to the port.</li> <li>Internal Network mode in which RXD from other ports are ANDed together. RXDSEL is ignored. INMMASK determines which RXD signals are ANDed together.</li> </ul> 0 Normal mode 1 Internal Network mode
7–0 INMMASK[7:0]	Internal Network Mode Mask. Bit mask that selects the ports from which the RXD signals are to be ANDed together for internal network mode. Bit 6 represents RXD from Port 7 and bit0 represents RXD from Port 1.  0 Includes RXDn for ANDing 1 Excludes RXDn from ANDing

**14.6.7 Port Timing Control Register 4 (AUDMUX\_PTCR4)**

## Port Timing Control Register for Port 4

Address: 21D\_8000h base + 18h offset = 21D\_8018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TFS_ DIR	TFSEL[3:0]				TCLKDIR	TCSEL[3:0]				RFS_ DIR	RFSEL[3:0]				RCLKDIR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RCSEL[3:0]				SYN	0										
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

## AUDMUX\_PTCR4 field descriptions

Field	Description
31 TFS_DIR	Transmit Frame Sync Direction Control. This bit sets the direction of the TXFS pin of the interface as an output or input. When set as an input, the TFSEL settings are ignored. When set as an output, the TFSEL settings determine the source port of the frame sync.  0 TXFS is an input. 1 TXFS is an output.
30–27 TFSEL[3:0]	Transmit Frame Sync Select. Selects the source port from which TXFS is sourced.  0xxx Selects TXFS from port. 1xxx Selects RXFS from port. x000 Port 1 x110 Port 7 x111 Reserved
26 TCLKDIR	Transmit Clock Direction Control. This bit sets the direction of the TXC pin of the interface as an output or input. When set as an input, the TCSEL settings are ignored. When set as an output, the TCSEL settings determine the source port of the clock.  0 TXC is an input. 1 TXC is an output.
25–22 TCSEL[3:0]	Transmit Clock Select. Selects the source port from which TXC is sourced.  0xxx Selects TXC from port. 1xxx Selects RXC from port. x000 Port 1 x110 Port 7 x111 Reserved
21 RFS_DIR	Receive Frame Sync Direction Control. This bit sets the direction of the RXFS pin of the interface as an output or input. When set as an input, the RFSEL settings are ignored. When set as an output, the RFSEL settings determine the source port of the frame sync.  0 RXFS is an input. 1 RXFS is an output.
20–17 RFSEL[3:0]	Receive Frame Sync Select. Selects the source port from which RXFS is sourced. RXFS can be sourced from TXFS and RXFS from other ports.  0xxx Selects TXFS from port. 1xxx Selects RXFS from port. x000 Port 1 x110 Port 7 x111 Reserved
16 RCLKDIR	Receive Clock Direction Control. This bit sets the direction of the RXC pin of the interface as an output or input. When set as an input, the RCSEL settings are ignored. When set as an output, the RCSEL settings determine the source port of the clock.  <b>NOTE:</b> RCLKDIR and SYN should not be changed at the same time.  0 RXC is an input. 1 RXC is an output.
15–12 RCSEL[3:0]	Receive Clock Select. Selects the source port from which RXC is sourced. RXC can be sourced from TXC and RXC from other ports.

*Table continues on the next page...*

**AUDMUX\_PTCR4 field descriptions (continued)**

Field	Description
	0xxx Selects TXC from port. 1xxx Selects RXC from port. x000 Port 1 x110 Port 7 x111 Reserved
11 SYN	Synchronous/Asynchronous Select. When SYN is set, synchronous mode is chosen and the transmit and receive sections use common clock and frame sync signals (that is, the port is a 4-wire interface). When SYN is cleared, asynchronous mode is chosen and separate clock and frame sync signals are used for the transmit and receive sections (that is, the port is a 6-wire interface).  <b>NOTE:</b> RCLKDIR and SYN should not be changed at the same time.  0 Asynchronous mode 1 Synchronous mode (default)
10–0 Reserved	This read-only field is reserved and always has the value 0.

**14.6.8 Port Data Control Register 4 (AUDMUX\_PDCR4)**

PDCR4 is the Port Data Control Register for Port 4.

Address: 21D\_8000h base + 1Ch offset = 21D\_801Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDSEL[2:0]			TXRXEN	0			MODE	INMMASK[7:0]							
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**AUDMUX\_PDCR4 field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–13 RXDSEL[2:0]	Receive Data Select. Selects the source port for the RXD data. RXDSEL is ignored if MODE = 1 (that is, Internal Network Mode is enabled).

*Table continues on the next page...*

**AUDMUX\_PDCR4 field descriptions (continued)**

Field	Description
	xxx Port number for RXD 000 Port 1 110 Port 7 111 Reserved
12 TXRXEN	Transmit/Receive Switch Enable. Swaps the transmit and receive signals. 0 No switch (Transmit Pin = Transmit, Receive Pin = Receive) 1 Switch (Transmit Pin = Receive, Receive Pin = Transmit)
11–9 Reserved	This read-only field is reserved and always has the value 0.
8 MODE	Mode Select. This field selects the mode in which the port is to operate. The modes of operation include the following: <ul style="list-style-type: none"> <li>Normal mode, in which the RXD from the port selected by RXDSEL is routed to the port.</li> <li>Internal Network mode in which RXD from other ports are ANDed together. RXDSEL is ignored. INMMASK determines which RXD signals are ANDed together.</li> </ul> 0 Normal mode 1 Internal Network mode
7–0 INMMASK[7:0]	Internal Network Mode Mask. Bit mask that selects the ports from which the RXD signals are to be ANDed together for internal network mode. Bit 6 represents RXD from Port 7 and bit0 represents RXD from Port 1. 0 Includes RXDn for ANDing 1 Excludes RXDn from ANDing

**14.6.9 Port Timing Control Register 5 (AUDMUX\_PTCR5)****Port Timing Control Register for Port 5**

Address: 21D\_8000h base + 20h offset = 21D\_8020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	TFS_DIR		TFSEL[3:0]				TCLKDIR	TCSEL[3:0]				RFS_DIR	RFSEL[3:0]				RCLKDIR
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	RCSEL[3:0]				SYN	0											
W																	
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	

## AUDMUX\_PTCR5 field descriptions

Field	Description
31 TFS_DIR	Transmit Frame Sync Direction Control. This bit sets the direction of the TXFS pin of the interface as an output or input. When set as an input, the TFSEL settings are ignored. When set as an output, the TFSEL settings determine the source port of the frame sync.  0 TXFS is an input. 1 TXFS is an output.
30–27 TFSEL[3:0]	Transmit Frame Sync Select. Selects the source port from which TXFS is sourced.  0xxx Selects TXFS from port. 1xxx Selects RXFS from port. x000 Port 1 x110 Port 7 x111 Reserved
26 TCLKDIR	Transmit Clock Direction Control. This bit sets the direction of the TXC pin of the interface as an output or input. When set as an input, the TCSEL settings are ignored. When set as an output, the TCSEL settings determine the source port of the clock.  0 TXC is an input. 1 TXC is an output.
25–22 TCSEL[3:0]	Transmit Clock Select. Selects the source port from which TXC is sourced.  0xxx Selects TXC from port. 1xxx Selects RXC from port. x000 Port 1 x110 Port 7 x111 Reserved
21 RFS_DIR	Receive Frame Sync Direction Control. This bit sets the direction of the RXFS pin of the interface as an output or input. When set as an input, the RFSEL settings are ignored. When set as an output, the RFSEL settings determine the source port of the frame sync.  0 RXFS is an input. 1 RXFS is an output.
20–17 RFSEL[3:0]	Receive Frame Sync Select. Selects the source port from which RXFS is sourced. RXFS can be sourced from TXFS and RXFS from other ports.  0xxx Selects TXFS from port. 1xxx Selects RXFS from port. x000 Port 1 x110 Port 7 x111 Reserved
16 RCLKDIR	Receive Clock Direction Control. This bit sets the direction of the RXC pin of the interface as an output or input. When set as an input, the RCSEL settings are ignored. When set as an output, the RCSEL settings determine the source port of the clock.  <b>NOTE:</b> RCLKDIR and SYN should not be changed at the same time.  0 RXC is an input. 1 RXC is an output.
15–12 RCSEL[3:0]	Receive Clock Select. Selects the source port from which RXC is sourced. RXC can be sourced from TXC and RXC from other ports.

*Table continues on the next page...*

**AUDMUX\_PTCR5 field descriptions (continued)**

Field	Description
	0xxx Selects TXC from port. 1xxx Selects RXC from port. x000 Port 1 x110 Port 7 x111 Reserved
11 SYN	Synchronous/Asynchronous Select. When SYN is set, synchronous mode is chosen and the transmit and receive sections use common clock and frame sync signals (that is, the port is a 4-wire interface). When SYN is cleared, asynchronous mode is chosen and separate clock and frame sync signals are used for the transmit and receive sections (that is, the port is a 6-wire interface).  <b>NOTE:</b> RCLKDIR and SYN should not be changed at the same time.  0 Asynchronous mode 1 Synchronous mode (default)
10–0 Reserved	This read-only field is reserved and always has the value 0.

**14.6.10 Port Data Control Register 5 (AUDMUX\_PDCR5)**

PDCR5 is the Port Data Control Register for Port 5.

Address: 21D\_8000h base + 24h offset = 21D\_8024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDSEL[2:0]			TXRXEN	0			MODE	INMMASK[7:0]							
W																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

**AUDMUX\_PDCR5 field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–13 RXDSEL[2:0]	Receive Data Select. Selects the source port for the RXD data. RXDSEL is ignored if MODE = 1 (that is, Internal Network Mode is enabled).

*Table continues on the next page...*

### AUDMUX\_PDCR5 field descriptions (continued)

Field	Description
	xxx Port number for RXD 000 Port 1 110 Port 7 111 Reserved
12 TXRXEN	Transmit/Receive Switch Enable. Swaps the transmit and receive signals. 0 No switch (Transmit Pin = Transmit, Receive Pin = Receive) 1 Switch (Transmit Pin = Receive, Receive Pin = Transmit)
11–9 Reserved	This read-only field is reserved and always has the value 0.
8 MODE	Mode Select. This field selects the mode in which the port is to operate. The modes of operation include the following: <ul style="list-style-type: none"> <li>Normal mode, in which the RXD from the port selected by RXDSEL is routed to the port.</li> <li>Internal Network mode in which RXD from other ports are ANDed together. RXDSEL is ignored. INMMASK determines which RXD signals are ANDed together.</li> </ul> 0 Normal mode 1 Internal Network mode
7–0 INMMASK[7:0]	Internal Network Mode Mask. Bit mask that selects the ports from which the RXD signals are to be ANDed together for internal network mode. Bit 6 represents RXD from Port 7 and bit0 represents RXD from Port 1. 0 Includes RXDn for ANDing 1 Excludes RXDn from ANDing

## 14.6.11 Port Timing Control Register 6 (AUDMUX\_PTCR6)

### Port Timing Control Register for Port 6

Address: 21D\_8000h base + 28h offset = 21D\_8028h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TFS_DIR			TFSEL[3:0]			TCLKDIR		TCSEL[3:0]			RFS_DIR	RFSEL[3:0]			RCLKDIR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RCSEL[3:0]				SYN	0										
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0



**AUDMUX\_PTCR6 field descriptions**

Field	Description
31 TFS_DIR	Transmit Frame Sync Direction Control. This bit sets the direction of the TXFS pin of the interface as an output or input. When set as an input, the TFSEL settings are ignored. When set as an output, the TFSEL settings determine the source port of the frame sync.  0 TXFS is an input. 1 TXFS is an output.
30–27 TFSEL[3:0]	Transmit Frame Sync Select. Selects the source port from which TXFS is sourced.  0xxx Selects TXFS from port. 1xxx Selects RXFS from port. x000 Port 1 x110 Port 7 x111 Reserved
26 TCLKDIR	Transmit Clock Direction Control. This bit sets the direction of the TXC pin of the interface as an output or input. When set as an input, the TCSEL settings are ignored. When set as an output, the TCSEL settings determine the source port of the clock.  0 TXC is an input. 1 TXC is an output.
25–22 TCSEL[3:0]	Transmit Clock Select. Selects the source port from which TXC is sourced.  0xxx Selects TXC from port. 1xxx Selects RXC from port. x000 Port 1 x110 Port 7 x111 Reserved
21 RFS_DIR	Receive Frame Sync Direction Control. This bit sets the direction of the RXFS pin of the interface as an output or input. When set as an input, the RFSEL settings are ignored. When set as an output, the RFSEL settings determine the source port of the frame sync.  0 RXFS is an input. 1 RXFS is an output.
20–17 RFSEL[3:0]	Receive Frame Sync Select. Selects the source port from which RXFS is sourced. RXFS can be sourced from TXFS and RXFS from other ports.  0xxx Selects TXFS from port. 1xxx Selects RXFS from port. x000 Port 1 x110 Port 7 x111 Reserved
16 RCLKDIR	Receive Clock Direction Control. This bit sets the direction of the RXC pin of the interface as an output or input. When set as an input, the RCSEL settings are ignored. When set as an output, the RCSEL settings determine the source port of the clock.  <b>NOTE:</b> RCLKDIR and SYN should not be changed at the same time.  0 RXC is an input. 1 RXC is an output.
15–12 RCSEL[3:0]	Receive Clock Select. Selects the source port from which RXC is sourced. RXC can be sourced from TXC and RXC from other ports.

*Table continues on the next page...*

**AUDMUX\_PTCR6 field descriptions (continued)**

Field	Description
	0xxx Selects TXC from port. 1xxx Selects RXC from port. x000 Port 1 x110 Port 7 x111 Reserved
11 SYN	Synchronous/Asynchronous Select. When SYN is set, synchronous mode is chosen and the transmit and receive sections use common clock and frame sync signals (that is, the port is a 4-wire interface). When SYN is cleared, asynchronous mode is chosen and separate clock and frame sync signals are used for the transmit and receive sections (that is, the port is a 6-wire interface).  <b>NOTE:</b> RCLKDIR and SYN should not be changed at the same time.  0 Asynchronous mode 1 Synchronous mode (default)
10–0 Reserved	This read-only field is reserved and always has the value 0.

**14.6.12 Port Data Control Register 6 (AUDMUX\_PDCR6)**

PDCR6 is the Port Data Control Register for Port 6.

Address: 21D\_8000h base + 2Ch offset = 21D\_802Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDSEL[2:0]			TXRXEN	0			MODE	INMMASK[7:0]							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**AUDMUX\_PDCR6 field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–13 RXDSEL[2:0]	Receive Data Select. Selects the source port for the RXD data. RXDSEL is ignored if MODE = 1 (that is, Internal Network Mode is enabled).

*Table continues on the next page...*

**AUDMUX\_PDCR6 field descriptions (continued)**

Field	Description
	xxx Port number for RXD 000 Port 1 110 Port 7 111 Reserved
12 TXRXEN	Transmit/Receive Switch Enable. Swaps the transmit and receive signals.  0 No switch (Transmit Pin = Transmit, Receive Pin = Receive) 1 Switch (Transmit Pin = Receive, Receive Pin = Transmit)
11–9 Reserved	This read-only field is reserved and always has the value 0.
8 MODE	Mode Select. This field selects the mode in which the port is to operate. The modes of operation include the following: <ul style="list-style-type: none"> <li>Normal mode, in which the RXD from the port selected by RXDSEL is routed to the port.</li> <li>Internal Network mode in which RXD from other ports are ANDed together. RXDSEL is ignored. INMMASK determines which RXD signals are ANDed together.</li> </ul> 0 Normal mode 1 Internal Network mode
7–0 INMMASK[7:0]	Internal Network Mode Mask. Bit mask that selects the ports from which the RXD signals are to be ANDed together for internal network mode. Bit 6 represents RXD from Port 7 and bit0 represents RXD from Port 1.  0 Includes RXDn for ANDing 1 Excludes RXDn from ANDing

**14.6.13 Port Timing Control Register 7 (AUDMUX\_PTCR7)****Port Timing Control Register for Port 7**

Address: 21D\_8000h base + 30h offset = 21D\_8030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TFS_DIR			TFSEL[3:0]			TCLKDIR		TCSEL[3:0]			RFS_DIR	RFSEL[3:0]			RCLKDIR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RCSEL[3:0]				SYN	0										
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

## AUDMUX\_PTCR7 field descriptions

Field	Description
31 TFS_DIR	Transmit Frame Sync Direction Control. This bit sets the direction of the TXFS pin of the interface as an output or input. When set as an input, the TFSEL settings are ignored. When set as an output, the TFSEL settings determine the source port of the frame sync.  0 TXFS is an input. 1 TXFS is an output.
30–27 TFSEL[3:0]	Transmit Frame Sync Select. Selects the source port from which TXFS is sourced.  0xxx Selects TXFS from port. 1xxx Selects RXFS from port. x000 Port 1 x110 Port 7 x111 Reserved
26 TCLKDIR	Transmit Clock Direction Control. This bit sets the direction of the TXC pin of the interface as an output or input. When set as an input, the TCSEL settings are ignored. When set as an output, the TCSEL settings determine the source port of the clock.  0 TXC is an input. 1 TXC is an output.
25–22 TCSEL[3:0]	Transmit Clock Select. Selects the source port from which TXC is sourced.  0xxx Selects TXC from port. 1xxx Selects RXC from port. x000 Port 1 x110 Port 7 x111 Reserved
21 RFS_DIR	Receive Frame Sync Direction Control. This bit sets the direction of the RXFS pin of the interface as an output or input. When set as an input, the RFSEL settings are ignored. When set as an output, the RFSEL settings determine the source port of the frame sync.  0 RXFS is an input. 1 RXFS is an output.
20–17 RFSEL[3:0]	Receive Frame Sync Select. Selects the source port from which RXFS is sourced. RXFS can be sourced from TXFS and RXFS from other ports.  0xxx Selects TXFS from port. 1xxx Selects RXFS from port. x000 Port 1 x110 Port 7 x111 Reserved
16 RCLKDIR	Receive Clock Direction Control. This bit sets the direction of the RXC pin of the interface as an output or input. When set as an input, the RCSEL settings are ignored. When set as an output, the RCSEL settings determine the source port of the clock.  <b>NOTE:</b> RCLKDIR and SYN should not be changed at the same time.  0 RXC is an input. 1 RXC is an output.
15–12 RCSEL[3:0]	Receive Clock Select. Selects the source port from which RXC is sourced. RXC can be sourced from TXC and RXC from other ports.

*Table continues on the next page...*

**AUDMUX\_PTCR7 field descriptions (continued)**

Field	Description
	0xxx Selects TXC from port. 1xxx Selects RXC from port. x000 Port 1 x110 Port 7 x111 Reserved
11 SYN	Synchronous/Asynchronous Select. When SYN is set, synchronous mode is chosen and the transmit and receive sections use common clock and frame sync signals (that is, the port is a 4-wire interface). When SYN is cleared, asynchronous mode is chosen and separate clock and frame sync signals are used for the transmit and receive sections (that is, the port is a 6-wire interface).  <b>NOTE:</b> RCLKDIR and SYN should not be changed at the same time.  0 Asynchronous mode 1 Synchronous mode (default)
10–0 Reserved	This read-only field is reserved and always has the value 0.

**14.6.14 Port Data Control Register 7 (AUDMUX\_PDCR7)**

PDCR7 is the Port Data Control Register for Port 7.

Address: 21D\_8000h base + 34h offset = 21D\_8034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RXDSEL[2:0]			TXRXEN	0			MODE	INMMASK[7:0]							
W																
Reset	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**AUDMUX\_PDCR7 field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–13 RXDSEL[2:0]	Receive Data Select. Selects the source port for the RXD data. RXDSEL is ignored if MODE = 1 (that is, Internal Network Mode is enabled).

*Table continues on the next page...*

**AUDMUX\_PDCR7 field descriptions (continued)**

Field	Description
	xxx Port number for RXD 000 Port 1 110 Port 7 111 Reserved
12 TXRXEN	Transmit/Receive Switch Enable. Swaps the transmit and receive signals. 0 No switch (Transmit Pin = Transmit, Receive Pin = Receive) 1 Switch (Transmit Pin = Receive, Receive Pin = Transmit)
11–9 Reserved	This read-only field is reserved and always has the value 0.
8 MODE	Mode Select. This field selects the mode in which the port is to operate. The modes of operation include the following: <ul style="list-style-type: none"> <li>• Normal mode, in which the RXD from the port selected by RXDSEL is routed to the port.</li> <li>• Internal Network mode in which RXD from other ports are ANDed together. RXDSEL is ignored. INMMASK determines which RXD signals are ANDed together.</li> </ul> 0 Normal mode 1 Internal Network mode
7–0 INMMASK[7:0]	Internal Network Mode Mask. Bit mask that selects the ports from which the RXD signals are to be ANDed together for internal network mode. Bit 6 represents RXD from Port 7 and bit0 represents RXD from Port 1. 0 Includes RXDn for ANDing 1 Excludes RXDn from ANDing

# Chapter 15

## Clock Controller Module (CCM)

### 15.1 Overview

The Clock Control Module (CCM) generates and controls clocks to the various modules in the design and manages low power modes. This module uses the available clock sources to generate the clock roots.

The Clock Controller Module controls the following functions:

- Uses the available clock sources to generate clock roots to various parts of the chip:
  - PLL1 also referenced as ARM PLL
  - PLL2 also referenced as System PLL
  - PLL3 also referenced as USB1 PLL
  - PLL4 also referenced as Audio PLL
  - PLL5 also referenced as Video PLL
  - PLL6 also referenced as ENET PLL
  - PLL7 also referenced as USB2 PLL (This PLL is only used by the USB UTM interface through a direct connection.)
- Uses programmable bits to control frequencies of the clock roots.
- Controls the low power mechanism.
- Provides control signals to LPCG for gating clocks.
- Provides handshake with SRC for reset performance.
- Provides handshake with GPC for support of DVFS and power gating operations.

#### 15.1.1 Features

The CCM includes these distinctive features:

- Provides root clock to SoC modules based on several source clocks.
- ARM core root clock is generated from a dedicated source clock.

- Includes separate dividers to control generation of core and bus root clocks (AXI, AHB, IPG).
- Includes separate dividers and clock source selectors for each serial root clock.
- Optional external clocks to bypass PLL clocks.
- Selects clock signals to output on CCM\_CLKO onto the pads for observability.
- Controllable registers are accessible via IP bus.
- Manages the Low Power Modes, namely RUN, WAIT and STOP. The gating of the peripheral clocks is programmable in RUN and WAIT modes.
- Manages frequency scaling procedure for ARM core clock by shifting between PLL sources, without loss of clocks.
- Manages frequency scaling procedure for peripheral root clock by programmable divider. The division is done on the fly without loss of clocks.
- Interface for the following IPs:
  - PLL - Interfaces for each PLL
  - LPCG - Low Power Clock Gating unit
  - SRC - System Reset Controller
  - GPC - General Power Controller

### 15.1.2 CCM Block Diagram



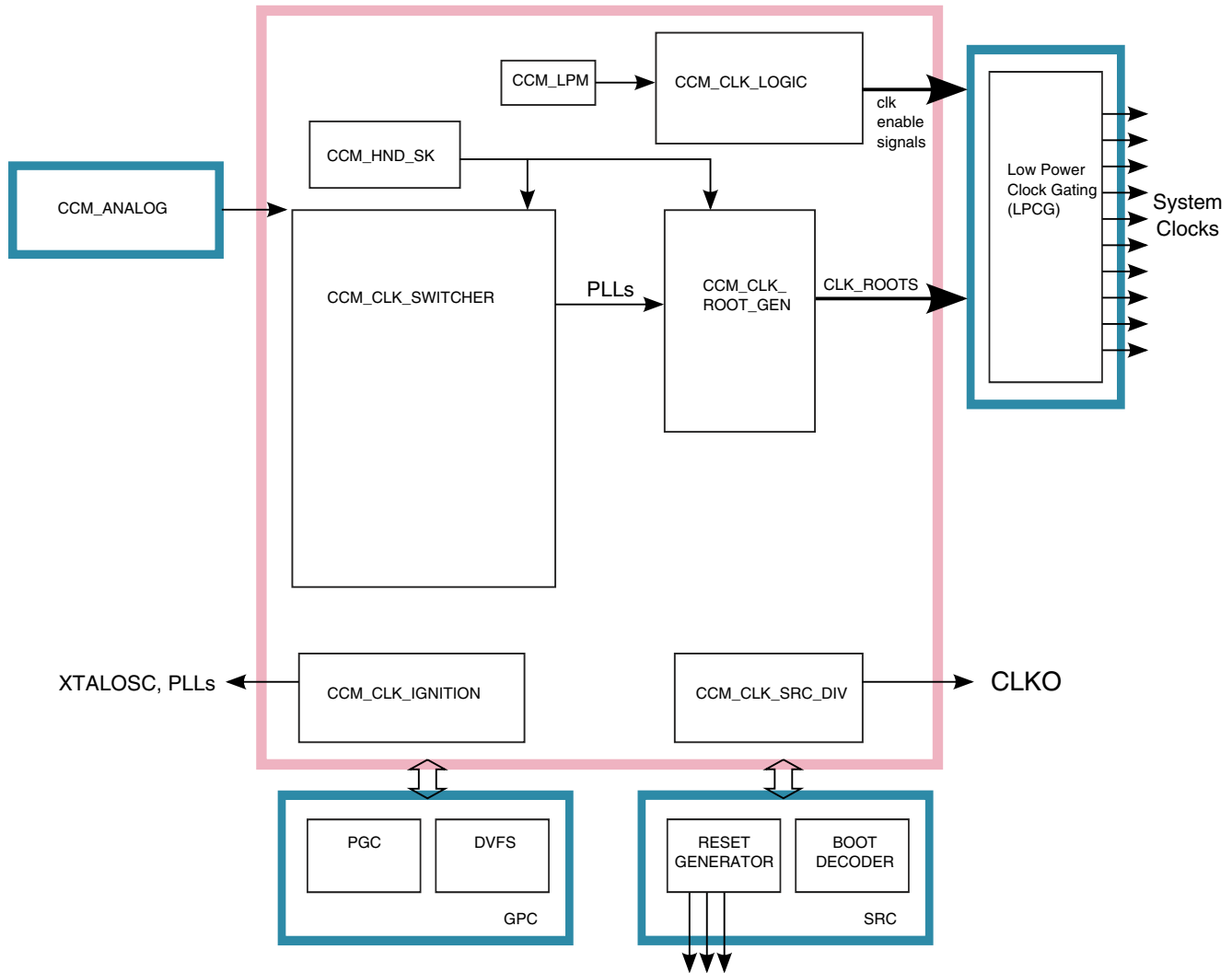


Figure 15-1. Block Diagram

CCM Contains the following sub-blocks:

Table 15-1. CCM Sub-blocks

Sub-block	Description
CCM_CLK_IGNITION	Manages the ignition process. This module starts its functionality once CCM comes out of reset. It manages the process that begins with starting the OSC, PLLs and finishes with creation of stable output root clocks after reset.
CCM_CLK_SWITCHER	Receives the clock outputs of the four PLLs, together with the bypass clocks for four PLLs , and generates three switcher clock outputs (pll1_sw_clk, pll2_sw_clk, pll3_sw_clk) for the CCM_CLK_ROOT_GEN sub-module.
CCM_CLK_ROOT_GEN	Receives the four main clocks (PLL1, PLL2, PLL3, PLL4) and generates the output root clocks.
CCM_CLK_LOGIC	Generates the clock enables. It generates the clock enable signals based on info from CCM_LPM and CCM_IP. The clock enables are used in LPCG to turn off and on the split clocks.

Table continues on the next page...

**Table 15-1. CCM Sub-blocks (continued)**

Sub-block	Description
CCM_LPM	Manages the low power modes of the IC.
CCM_CLK_SRC_DIV	Passes CCM_CLKO output clock for observability. Output clock is connected to the pad and can provide support for testing.
CCM_HND_SK	Manages the handshake when changing root clock dividers that require handshake, and manages the frequency change in case of dvfs scenario.

## 15.2 External Signals

The following table describes the external signals of CCM:

**Table 15-2. CCM External Signals**

Signal	Description	Pad	Mode	Direction
CCM_CLKO	Observability clock output	PWM1	ALT1	O
CCM_PMIC_READY	Signal coming from PMIC to indicate that the voltage started to change as result of change in CCM_PMIC_VSTBY_REQ	FEC_REF_CLK	ALT4	I
		LCD_RESET	ALT6	
		REF_CLK_24M	ALT4	
		SD1_DAT7	ALT3	
CCM_PMIC_VSTBY_REQ	Goes to PMIC_VSTBY_REQ pin, which notifies external power management IC to move from functional voltage to standby voltage.	PMIC_STBY_REQ	No Muxing (ALT0)	I

## 15.3 CCM Clock Tree

The figure found here shows the clock tree configuration and clock roots for CCM.

For detailed sub-block information, please see:

- [Clock Switcher](#)
- [Clock Root Generator](#)
- [Low Power Clock Gating module \(LPCG\)](#)
- [System Clocks](#)

### NOTE

The default frequency values (in MHz) for the PLLs and PFDs is the maximum allowed frequency. The PLL and PFD control registers should not be programmed to exceed these values.

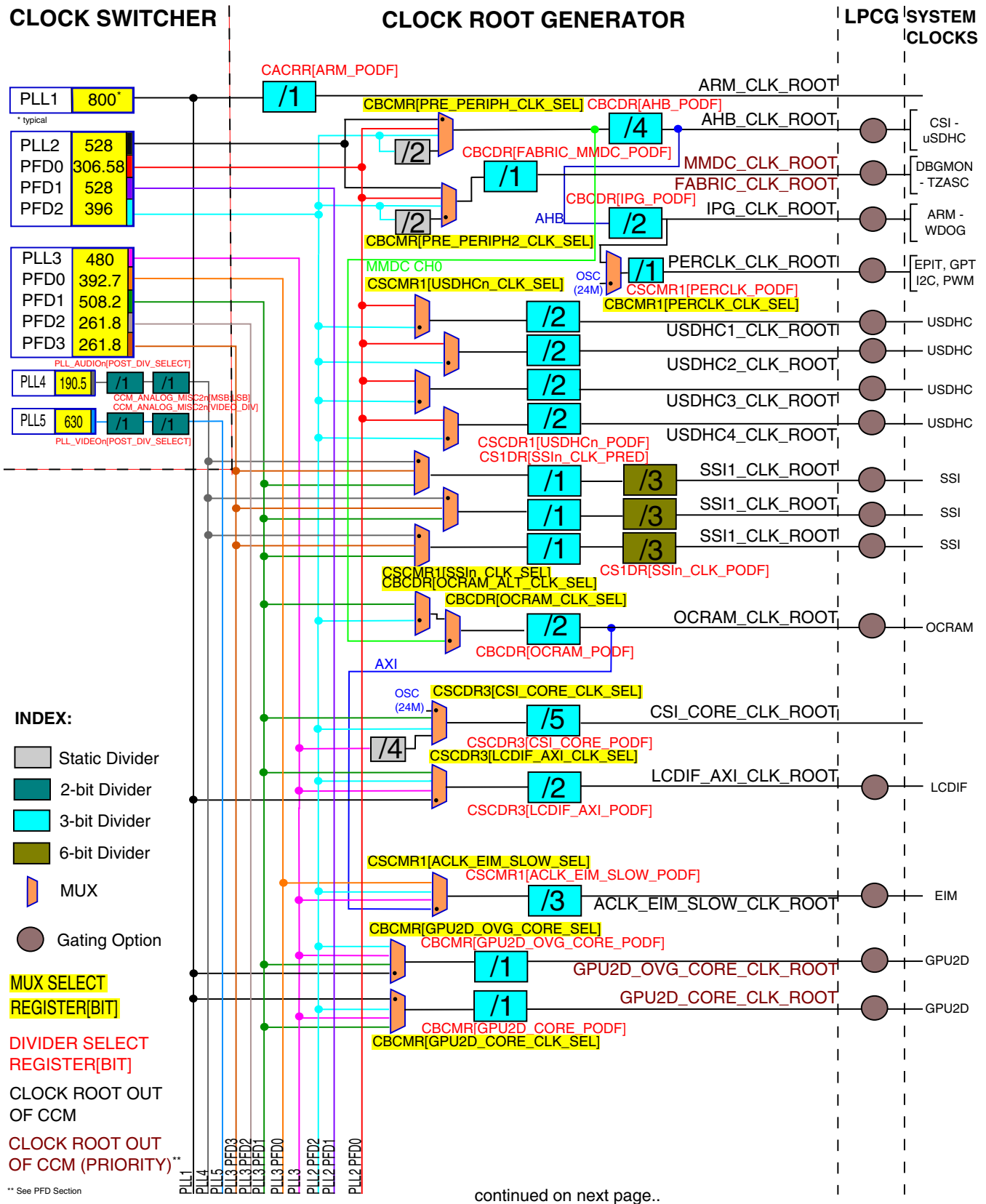


Figure 15-2. Clock Tree - Part 1



### Figure 15-3. Clock Tree - Part 2

## 15.4 System Clocks

The table found here shows the CCM output clocks' system-level connectivity.

The gating option in the table can either be CGR bit or clock enable from the block itself. Applicable override bits are also shown.

Block Instance	Clock	Clock Root	Gating CCM_CCGR bits	Override CCM_CMEOR bits
AIPSTZ <i>n</i>	hclk	ahb_clk_root	aips_tz1_clk_enable	
AUDMUX	ipg_clk_s	ipg_clk_root		

*Table continues on the next page...*

Block Instance	Clock	Clock Root	Gating CCM_CCGR bits	Override CCM_CMEOR bits
CCM	ana_apb_clk	ipg_clk_root		
	ipg_clk_s	ipg_clk_root		
	ipg_clk	ipg_clk_root		
CSI	csi_hclk	ahb_clk_root		
	ipg_clk	ipg_clk_root		
	ipg_clk_s	ipg_clk_root		
	ipg_clk_s_raw	ipg_clk_root		
CSU	ap_ckil_clk	ckil_sync_clk_root		
	ipg_clk_s	ipg_clk_root		
DBGMON	clk	ipg_clk_root		
	apb_axi_ACLK	ipg_clk_root		
	axi_ACLK	mmdc_axi_clk_root		
DCP	clk	ahb_clk_root		
	mem_clk	ahb_clk_root		
ECSPIn	ipg_clk	ipg_clk_root	ecspi1_clk_enable	
	ipg_clk_32k	ckil_sync_clk_root		
	ipg_clk_per	ecspi_clk_root	ecspi1_clk_enable	
	ipg_clk_s	ipg_clk_root		
EIM	aclk	aclk_eim_slow_clk_root	emi_slow_clk_enable	
	aclk_slow	aclk_eim_slow_clk_root	emi_slow_clk_enable	
	ipg_clk_s	ipg_clk_root		
	aclk_exsc	aclk_eim_slow_clk_root	emi_slow_clk_enable	
EPDC	aclk	epdc_axi_clk_root	epdc_axi_clk_enable	
	pixclk	epdc_pix_clk_root	epdc_pix_clk_enable	
EPITn	ipg_clk	perclk_clk_root	epit1_clk_enable	mod_en_ov_epit
	ipg_clk_32k	ckil_sync_clk_root		
	ipg_clk_highfreq	perclk_clk_root	epit1_clk_enable	
	ipg_clk_s	perclk_clk_root		
FEC	fec_clk	ipg_clk_root		
	fec_hclk	ahb_clk_root		
GPU2D	gc320_sec.mst_hclk	ahb_clk_root	gpu_clk_enable	mod_en_ov_gpu2d
	gc355_sec.mst_hclk	ahb_clk_root	gpu_clk_enable	mod_en_ov_gpu2d
	gpu2d.ACLK	mmdc_axi_clk_root	gpu_clk_enable	mod_en_ov_gpu2d
	gc320_clk2x	gc320_clk_root	gpu_clk_enable	mod_en_ov_gpu2d
	gc320_HCLK	ahb_clk_root	gpu_clk_enable	mod_en_ov_gpu2d
	gc355_V2D_clk2x	gc355_clk_root	gpu_clk_enable	mod_en_ov_gpu2d
	gc355_V2D_HCLK	ahb_clk_root	gpu_clk_enable	mod_en_ov_gpu2d
	sms_wrck	wrck_clk_root		

Table continues on the next page...

## System Clocks

Block Instance	Clock	Clock Root	Gating CCM_CCGR bits	Override CCM_CMEOR bits
GPC	ipg_clk	ipg_clk_root		
	ipg_clk_s	ipg_clk_root		
	pgc_clk	ipg_clk_root		
	sys_clk	ipg_clk_root		
GPION	ipg_clk_s	ipg_clk_root		
GPT	gpt.ipg_clk	perclk_clk_root	gpt_clk_enable	mod_en_ov_gpt
	ipg_clk_32k	ckil_sync_clk_root		
	ipg_clk_highfreq	perclk_clk_root	gpt_serial_clk_enable	
	ipg_clk_s	perclk_clk_root		
I2Cn	ipg_clk_patref	perclk_clk_root	i2c1_serial_clk_enable	
	ipg_clk_s	perclk_clk_root		
IOMUXC	ipt_clk_io	lcdif_pix_clk_root	iomux_ipt_clk_io_enable	
	ipg_clk_s	ipg_clk_root		
KPP	ipg_clk_32k	ckil_sync_clk_root		
	ipg_clk_s	ipg_clk_root		
LCDIF	apb_clk	lcdif_axi_clk_root	lcdif_axi_clk_enable	
	pix_clk	lcdif_pix_clk_root	lcdif_pix_clk_enable	
MMDC	aclk_fast_core_p0	mmdc_axi_clk_root	mmdc_core_aclk_fast_core_p0_enable	
	aclk_fast_phy_p0	mmdc_axi_clk_root	mmdc_core_aclk_fast_core_p0_enable	
	clk32	xtalosc	mmdc_core_ipg_clk_p1_enable	
	ipg_clk_p0	ipg_clk_root	mmdc_core_ipg_clk_p0_enable	
OCOTP	ipg_clk	ipg_clk_root	iim_clk_enable	
	ipg_clk_s	ipg_clk_root		
OCRAM	clk	axi_clk_root	ocram_clk_enable	
	l2_clk	axi_clk_root	ocram_clk_enable	
	mem_clk	axi_clk_root	ocram_clk_enable	
	rdata_sel_clk	axi_clk_root	ocram_clk_enable	
PWMn	ipg_clk	perclk_clk_root	pwm1_clk_enable	
	ipg_clk_32k	ckil_sync_clk_root		
	ipg_clk_highfreq	perclk_clk_root	pwm1_clk_enable	
	ipg_clk_s	perclk_clk_root		
PXP	clk	pxp_axi_clk_root	pxp_axi_clk_enable	
RNGB	ipg_clk	ipg_clk_root		
	ipg_clk_s	ipg_clk_root		

Table continues on the next page...

Block Instance	Clock	Clock Root	Gating CCM_CCGR bits	Override CCM_CMEOR bits
ROMCP	rom_96k.rom_CLK	ahb_clk_root	rom_clk_enable	
	hclk	ahb_clk_root	rom_clk_enable	
	hclk_reg	ipg_clk_root	rom_clk_enable	
	romcp_sec.mst_hclk	ahb_clk_root	rom_clk_enable	
SDMA	ips_hostctrl_clk	ipg_clk_root	sdma_clk_enable	
	sdma_ap_ahb_clk	ahb_clk_root	sdma_clk_enable	
	sdma_core_clk	ipg_clk_root	sdma_clk_enable	
	tck	sjc_tck_fixme		
	sdma_events_sync.clk	ahb_clk_root	sdma_clk_enable	
SNVS	snvs_hp_wrapper.ipg_clk	ipg_clk_root		
	snvs_hp_wrapper.ipg_clk_s	ipg_clk_root		
	snvs_hp_wrapper.ipg_hp_rtc_clk	ckil_sync_clk_root		
	snvs_lp_wrapper.ipg_clk	ipg_clk_root		
	snvs_lp_wrapper.ipg_clk_s	ipg_clk_root		
SPBA	ipg_clk	ipg_clk_root	spba_clk_enable	
	ipg_clk_s	ipg_clk_root	spba_clk_enable	
SPDIF	gclkw_t0	ipg_clk_root	spdif_clk_enable	
	ipg_clk_s	ipg_clk_root		
	tx_clk	spdif0_clk_root	spdif_clk_enable	
SRC	ipg_clk	ipg_clk_root		
	src_ipg_clk_s	ipg_clk_root		
	wrck	wrck_clk_root		
SSI <sub>n</sub>	ccm_ssi_clk	ssi1_clk_root	ssi1_clk_enable	
	ipg_clk	ipg_clk_root	ssi1_clk_enable	
	ipg_clk_s	ipg_clk_root		
TZASC	aclk	mmdc_axi_clk_root	ipsync_ip2apb_tzasc1_ipg_master_clk_enable	
UART <sub>n</sub>	ipg_clk	ipg_clk_root	uart_clk_enable	
	ipg_clk_s	ipg_clk_root		
	ipg_perclk	uart_clk_root	uart_serial_clk_enable	
USB	ipg_ahb_clk	ahb_clk_root	usboh3_clk_enable	
	ipg_clk_32khz	ckil_sync_clk_root		
	ipg_clk_s	ipg_clk_root		
	ipg_clk_s_pl301	ipg_clk_root	usboh3_clk_enable	
	test_clk_240m	ipg_clk_root	usboh3_clk_enable	
	test_clk_480m	ipg_clk_root	usboh3_clk_enable	
	test_clk_60m	ipg_clk_root	usboh3_clk_enable	

Table continues on the next page...

## Functional Description

Block Instance	Clock	Clock Root	Gating CCM_CCGR bits	Override CCM_CMEOR bits
USDHC $n$	hclk	ahb_clk_root	usdhc1_clk_enable	mod_en_ov_usdhc
	ipg_clk	ipg_clk_root	usdhc1_clk_enable	mod_en_ov_usdhc
	ipg_clk_perclk	usdhc1_clk_root	usdhc1_clk_enable	mod_en_ov_usdhc
	ipg_clk_s	ipg_clk_root		mod_en_ov_usdhc
WDOG $n$	ipg_clk	ipg_clk_root		
	ipg_clk_32k	ckil_sync_clk_root		
	ipg_clk_s	ipg_clk_root		

**Table 15-3. System Clock Frequency Values**

Clock Root	Default Frequency (MHz)	Maximum Frequency (MHz)
ARM_CLK_ROOT	792	
PERCLK_CLK_ROOT	66	66
AHB_CLK_ROOT	132	133
IPG_CLK_ROOT	66	133
MMDC_CLK_ROOT	198	528
USDHC $n$ _CLK_ROOT	198	400
SSIn_CLK_ROOT	63.5	66
OCRAM_CLK_ROOT	264	540
CSI_CORE_CLK_ROOT	4.8	540
LCDIF_AXI_CLK_ROOT	264	540
ACLK_EIM_SLOW_CLK_ROOT	264	480
GPU2D_OVG_CORE_CLK_ROOT	528	594
GPU2D_CORE_CLK_ROOT	528	720
PXP_AXI_CLK_ROOT	87.3	540
EPDC_AXI_CLK_ROOT	87.3	540
LCDIF_PIX_CLK_ROOT	180	540
EPDC_PIX_CLK_ROOT	56.5	540
SPDIF0_CLK_ROOT	30	60
SPDIF1_CLK_ROOT	30	60
EXTERN_AUDIO_CLK_ROOT	30	400
ECSPI_CLK_ROOT	4	60
UART_CLK_ROOT	4	80

## 15.5 Functional Description

This section provides a complete functional description of the block.



## 15.5.1 Clock Generation

### 15.5.1.1 External Low Frequency Clock - CKIL

The chip can use a 32 kHz or 32.768 kHz crystal as the external low-frequency source (XTALOSC). Throughout this chapter, the low-frequency crystal is referred to as the 32 kHz crystal.

This clock source should always be active when the chip is powered on. The 32 kHz entering the CCM are referred to as CKIL. CKIL is synchronized to IPG\_CLK and supplied to modules that need it.

#### 15.5.1.1.1 CKIL synchronizing to IPG\_CLK

CKIL is synchronized to ipg\_clk when the system is in functional mode. When the system is in STOP mode (when there is no IPG\_CLK) the CKIL synchronizer is bypassed, and raw CKIL is supplied to the system.

### 15.5.1.2 External High Frequency Clock - CKIH and internal oscillator

The chip uses an internal oscillator to generate the reference clock (OSC). The internal oscillator is connected to the external crystal (XTALOSC) which generates the 24 MHz reference clock.

### 15.5.1.3 PLL reference clock

There are several PLLs in this chip.

PLL1 - ARM PLL (typical functional frequency 1 GHz)

PLL2 - System PLL (functional frequency 528 MHz)

PLL3 - USB1 PLL (functional frequency 480 MHz)

PLL4 - Audio PLL

PLL5 - Video PLL

PLL6 - ENET PLL

PLL7 - USB2 PLL (functional frequency 480 MHz)

Some of the PLLs are described in the sections below. See [CCM Analog Memory Map/ Register Definition](#) for register information.

#### 15.5.1.3.1 ARM PLL

This PLL synthesizes a low jitter clock from a 24 MHz reference clock. The clock output frequency for this PLL ranges from 650 MHz to 1.3 GHz. Note that the PLL's upper frequency range exceeds the maximum frequency supported by the system. The output frequency is selected by a 7-bit register field CCM\_ANALOG\_PLL\_ARM[DIV\_SELECT].

$$\text{PLL output frequency} = \text{Fref} * \text{DIV\_SEL}/2$$

#### 15.5.1.3.2 USB PLLs

These PLLs synthesize a low jitter clock from the 24 MHz reference clock. USB1 PLL has 4 frequency-programmable PFD (phase fractional divider) outputs.

The output frequency of USB1 PLL is 480 MHz. Even though USB1 PLL has a DIV\_SELECT register field, this PLL should always be set to 480 MHz in normal operation. USB2 PLL is only used by the USB UTM interface through a direct connection.

#### 15.5.1.3.3 System PLL

This PLL synthesizes a low jitter clock from the 24 MHz reference clock. The PLL has one output clock, plus 3 PFD outputs.

Although this PLL does have a DIV\_SELECT register field, it is intended that this PLL will only be run at the default frequency of 528 MHz.

#### 15.5.1.3.4 Audio / Video PLL

The audio PLL and video PLL each synthesize a low jitter clock from a 24 MHz reference clock. The clock output frequency range for this PLL is from 650 MHz to 1.3 GHz. It has a Fractional-N synthesizer.

There are /1, /2, /4, /8, /16 post dividers for the Video PLL and /1, /2, /4, /8, /16 post dividers for the Audio PLL. The output frequency can be set by programming the fields in the CCM\_ANALOG\_PLL\_AUDIO, CCM\_ANALOG\_PLL\_VIDEO, and CCM\_ANALOG\_MISC2 register sets according to the following equation.

$$\text{PLL output frequency} = \text{Fref} * (\text{DIV\_SELECT} + \text{NUM/DENOM})$$

### 15.5.1.3.5 Ethernet PLL

This PLL synthesizes a low jitter clock from the 24 MHz reference clock.

The PLL outputs a 50 MHz clock. The reference clocks generated by this PLL are:

- fixed 125 MHz
- fixed 100 MHz
- Ref\_ethernet, which is configurable based on the PLL\_ENET[1:0] register field.

### 15.5.1.4 Phase Fractional Dividers (PFD)

There are several PFD outputs from the System PLL and USB1 PLL.

Each PFD output generates a fractional multiplication of the associated PLL's VCO frequency. Where the output frequency is equal to  $F_{vco} \times 18/N$ , N can range from 12-35. The PFDs allow for clock frequency changes without forcing the relock of the root PLL. This feature is useful in support of dynamic voltage and frequency scaling (DVFS). See [CCM Analog Memory Map/Register Definition](#).

When the related PLL is powered up from the power down state or made to go through a relock cycle due to PLL reprogramming, it is required that the related PFDx\_CLKGATE bit in CCM\_ANALOG\_PFD\_480n or CCM\_ANALOG\_PFD\_528n, be cycled on and off (1 to 0) after PLL lock. The PFDs can be in the clock gated state during PLL relock but must be un-clock gated only after lock is achieved. See the engineering bulletin, *Configuration of Phase Fractional Dividers (EB790)* at [www.freescale.com](http://www.freescale.com) for procedure details.

The chip has 7 PFDs, of which one is fixed-frequency. The others are assigned to a prioritized clock root. Prioritized clock roots are set by software and have priority over other shared clock roots. Non-prioritized clock roots should only select a particular PFD as their source clock if the frequency is appropriate or not selected by the main prioritized clock root. All clock roots have been designed with sufficient clock source selections to avoid a mutually-exclusive condition in which two clock sources must use a PFD simultaneously, but require incompatible frequencies.

The following table shows the clock root to PFD priority assignments.

#### NOTE

PLL2\_PFD2 should be programmed at a fixed frequency of 396MHz.

Root Clock with Priority	PFD Assigned
FABRIC_CLK_ROOT/MMDC_CLK_ROOT	PLL2_PFD0

*Table continues on the next page...*

## Functional Description

Root Clock with Priority	PFD Assigned
EPDC_PIX_CLK	PLL2_PFD1
LCDIF_PIX_CLK	PLL3_PFD0
GPU2D_OVG_CORE_CLK_ROOT/GPU2D_CORE_CLK_ROOT	PLL3_PFD1
EPDC_AXI_CLK	PLL3_PFD2
PXP_AXI_CLK	PLL3_PFD3

### 15.5.1.5 CCM internal clock generation

The clock generation is comprised of two sub-modules:

CCM\_CLK\_SWITCHER

CCM\_CLK\_ROOT\_GEN

#### 15.5.1.5.1 Clock Switcher

The Clock Switcher (CCM\_CLK\_SWITCHER) sub-module receives the PLL output clocks and the PLL bypass clocks.

[Figure 15-4](#) describes the generation of the three switcher clocks.

The figure also includes the Frequency Switch Control sub-module responsible for frequency change.

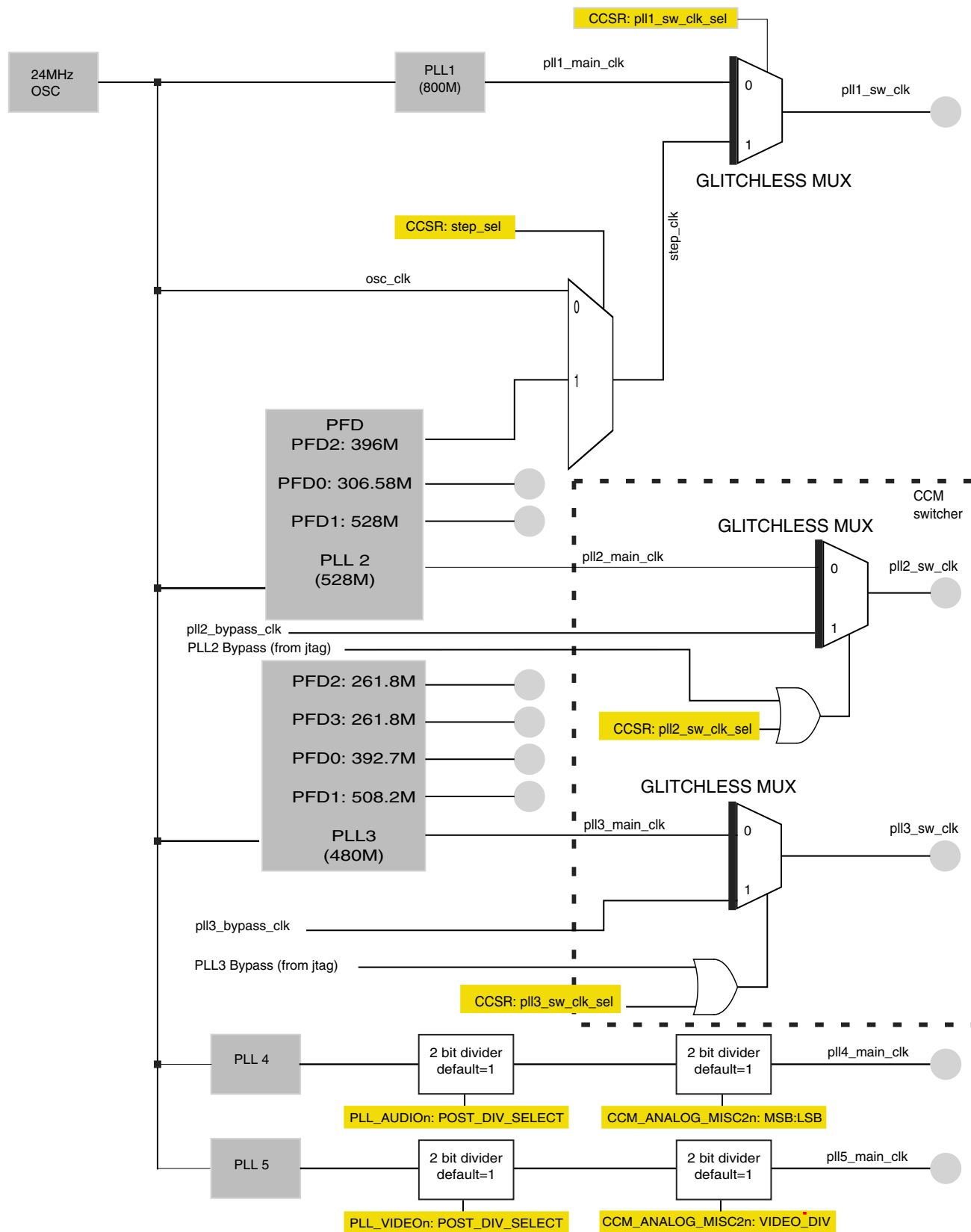


Figure 15-4. Switcher clock generation

### 15.5.1.5.2 PLL bypass procedure

In addition to PLL bypass options in CCM\_ANALOG module, switcher and clk\_root\_gen sub-modules includes capability for each of the PLL clocks to be bypassed with an external bypass clk, and in this way supply to the pll\_sw\_clk outputs an external bypass clock.

### 15.5.1.5.3 PLL clock change

In order to modify or stop the clock output of a specific PLL, all the clocks generated from the current PLL must be transitioned to the new PLL whose frequency is not being modified.

For clocks which can't be stopped (core and bus clocks), this should be done via the glitchless mux. Before changing the PLL setting, power it down. Power up the PLL after the change. See [Disabling / Enabling PLLs](#) for more information.

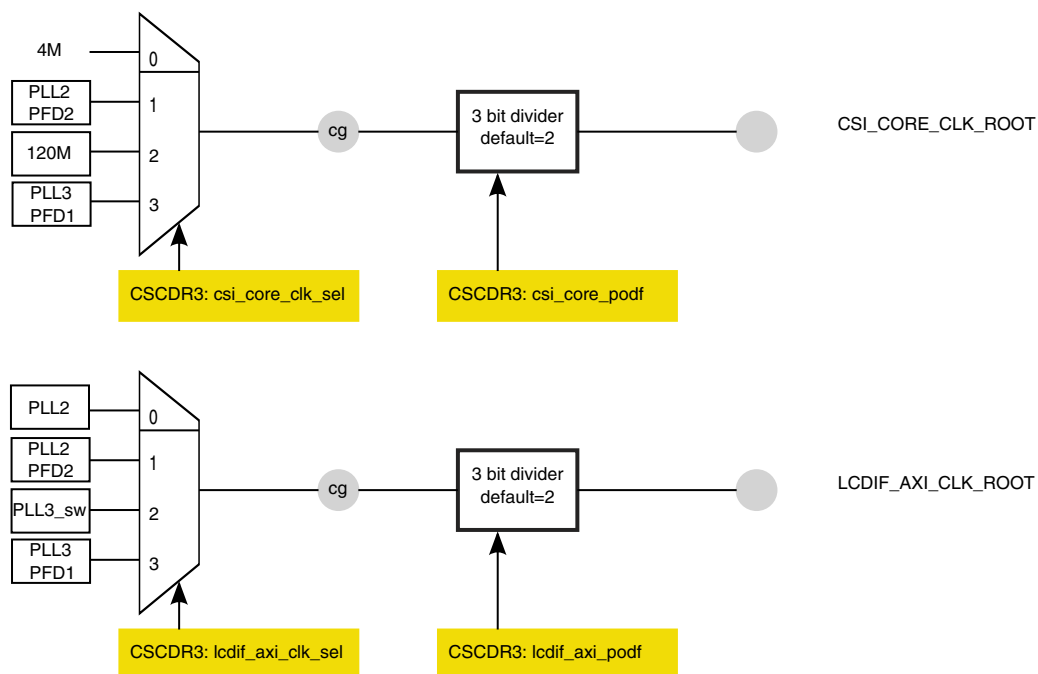
### 15.5.1.5.4 Clock Root Generator

The Clock Root Generator (CCM\_CLK\_ROOT\_GEN) sub-module generates the root clocks to be delivered to LPCG.

The following figures describe clock generation. The frequencies in parantheses are the default typical frequencies.



## Functional Description





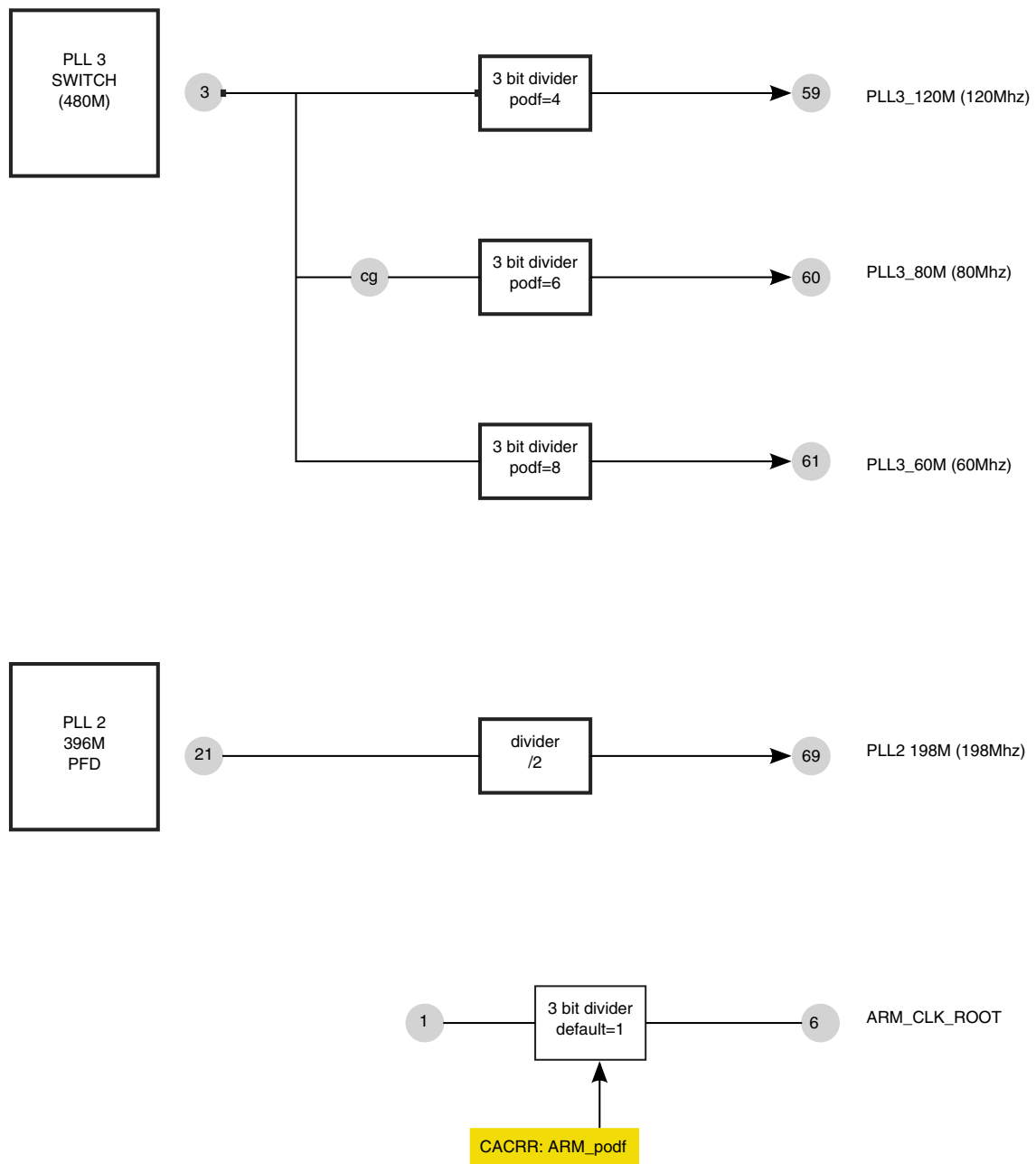


Figure 15-6. AXI clocks generation

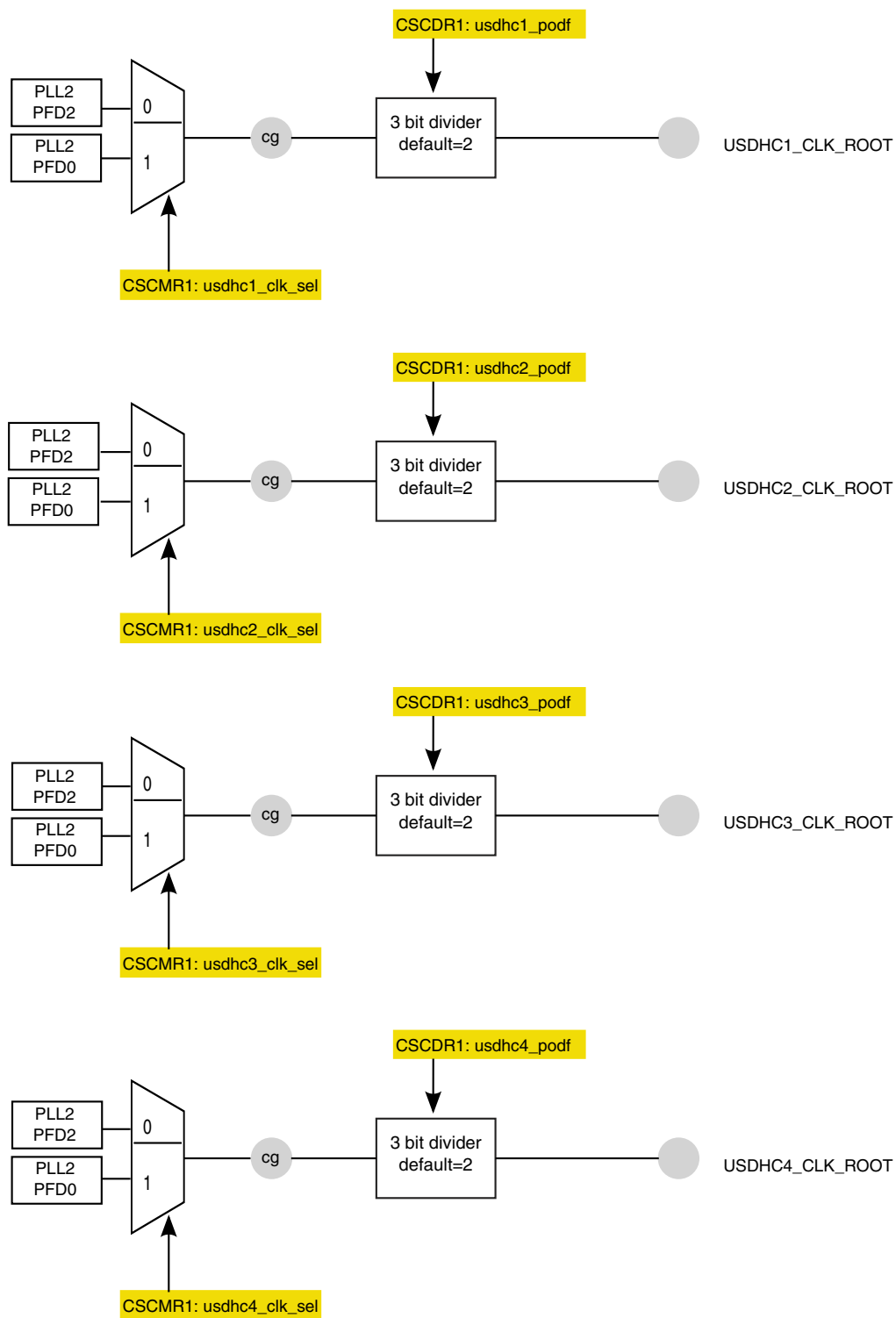
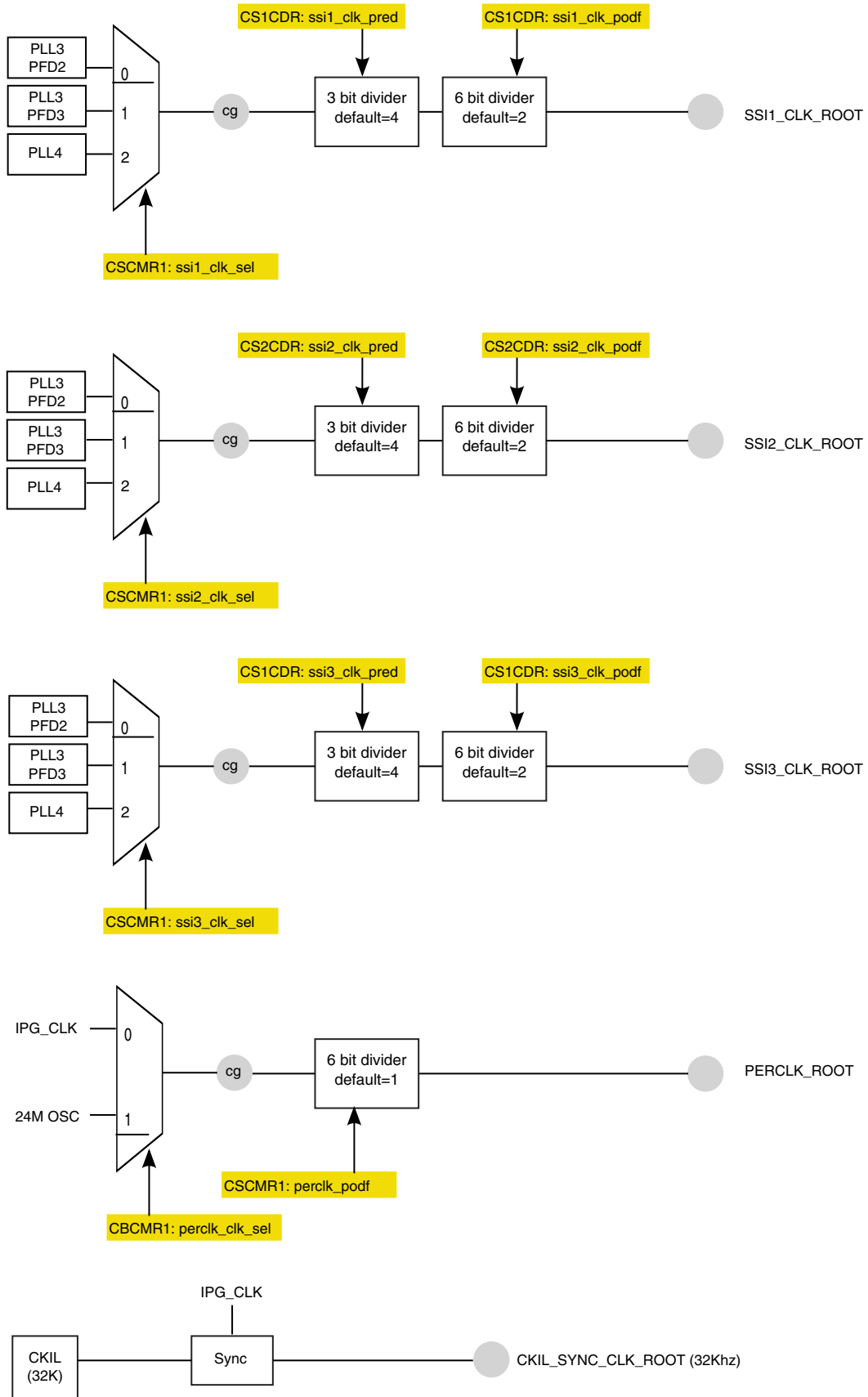
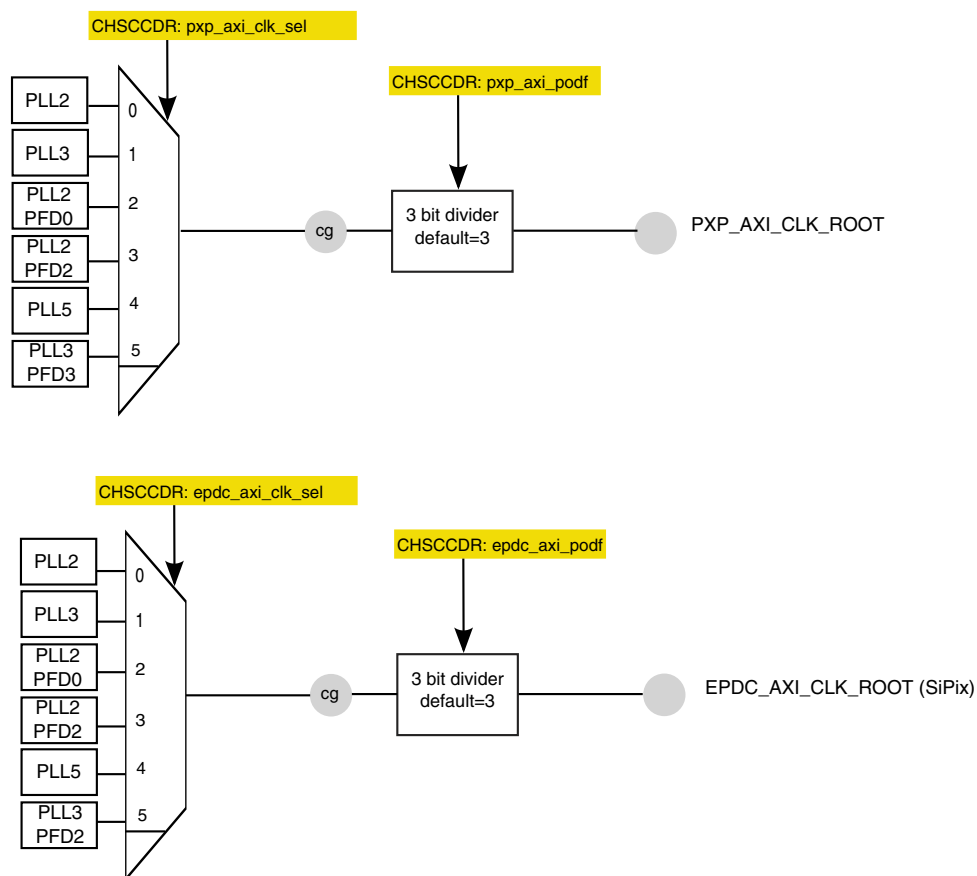
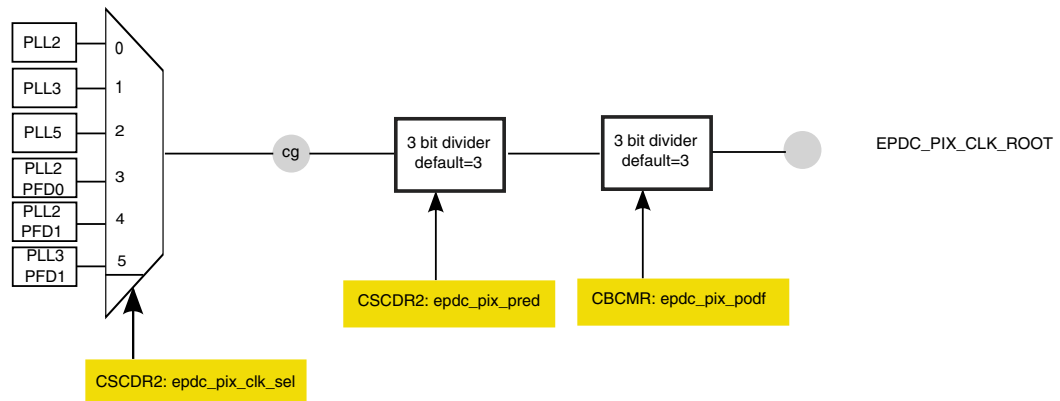
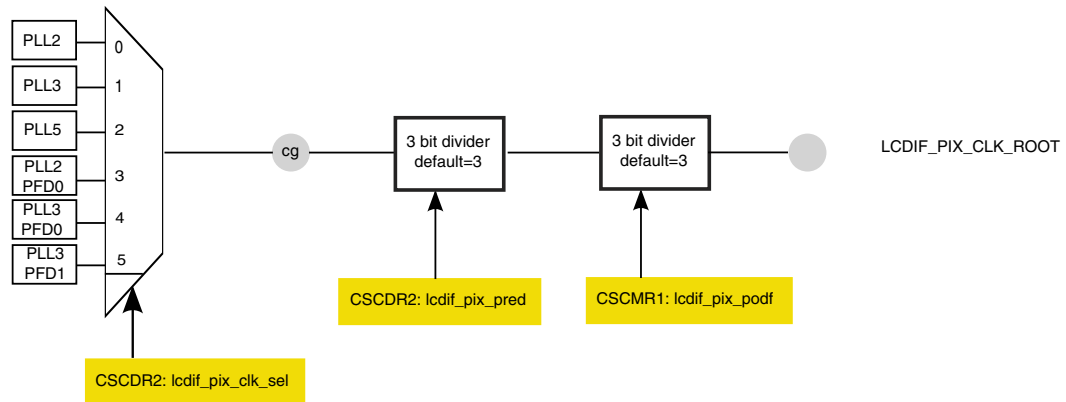
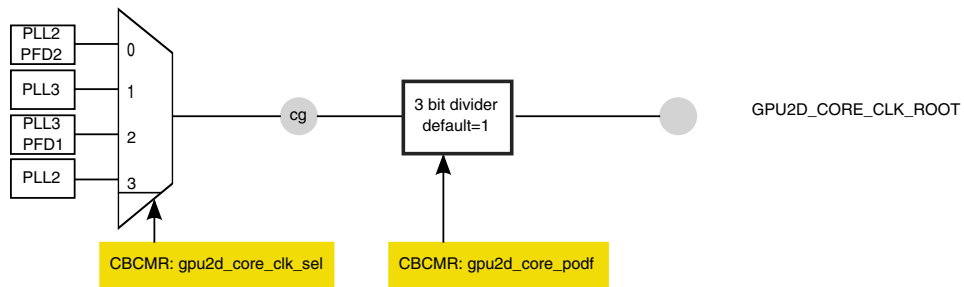
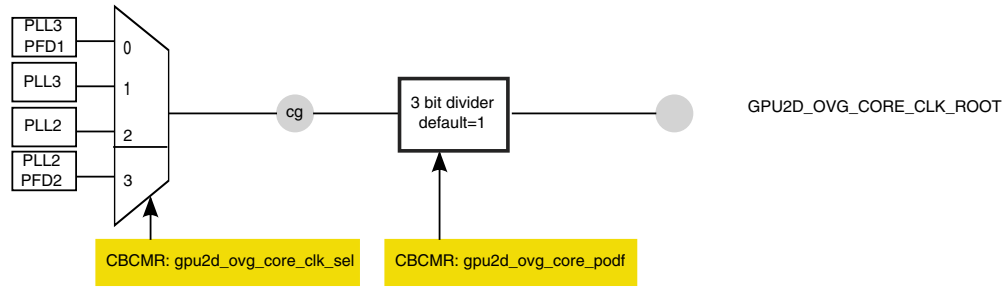


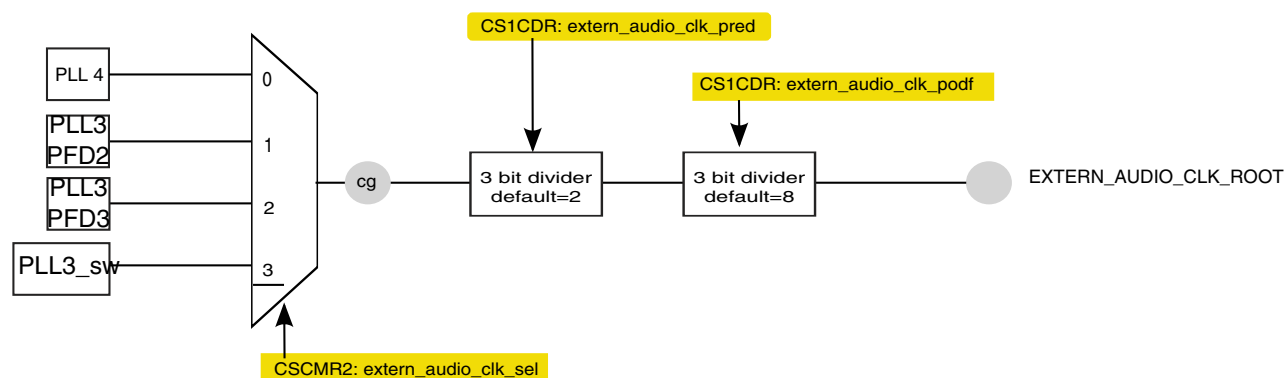
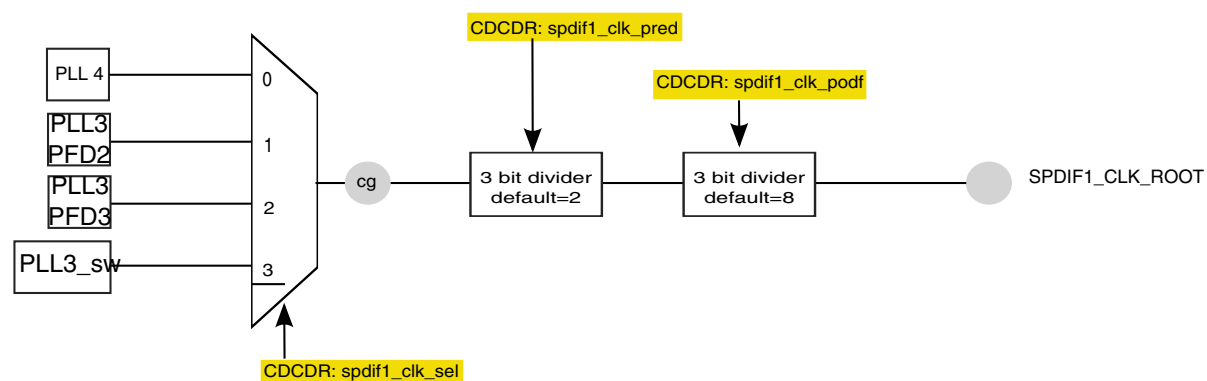
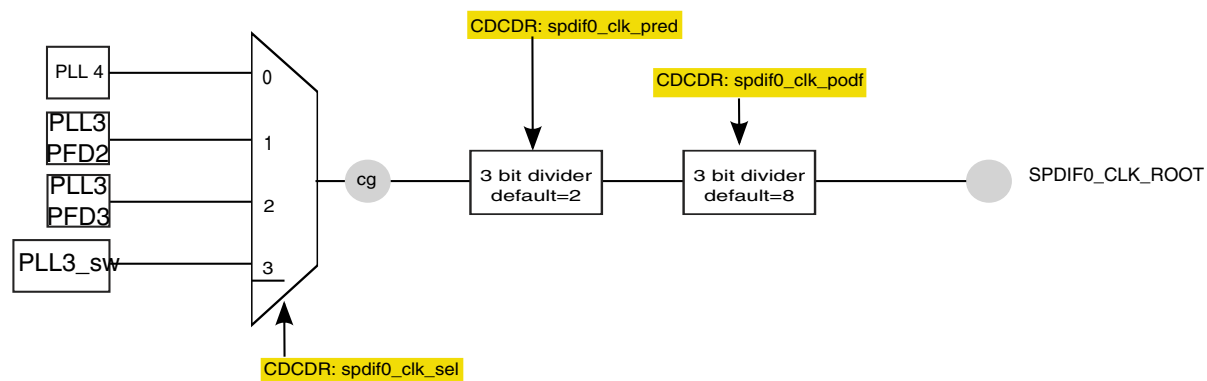
Figure 15-7. Serial clock generation

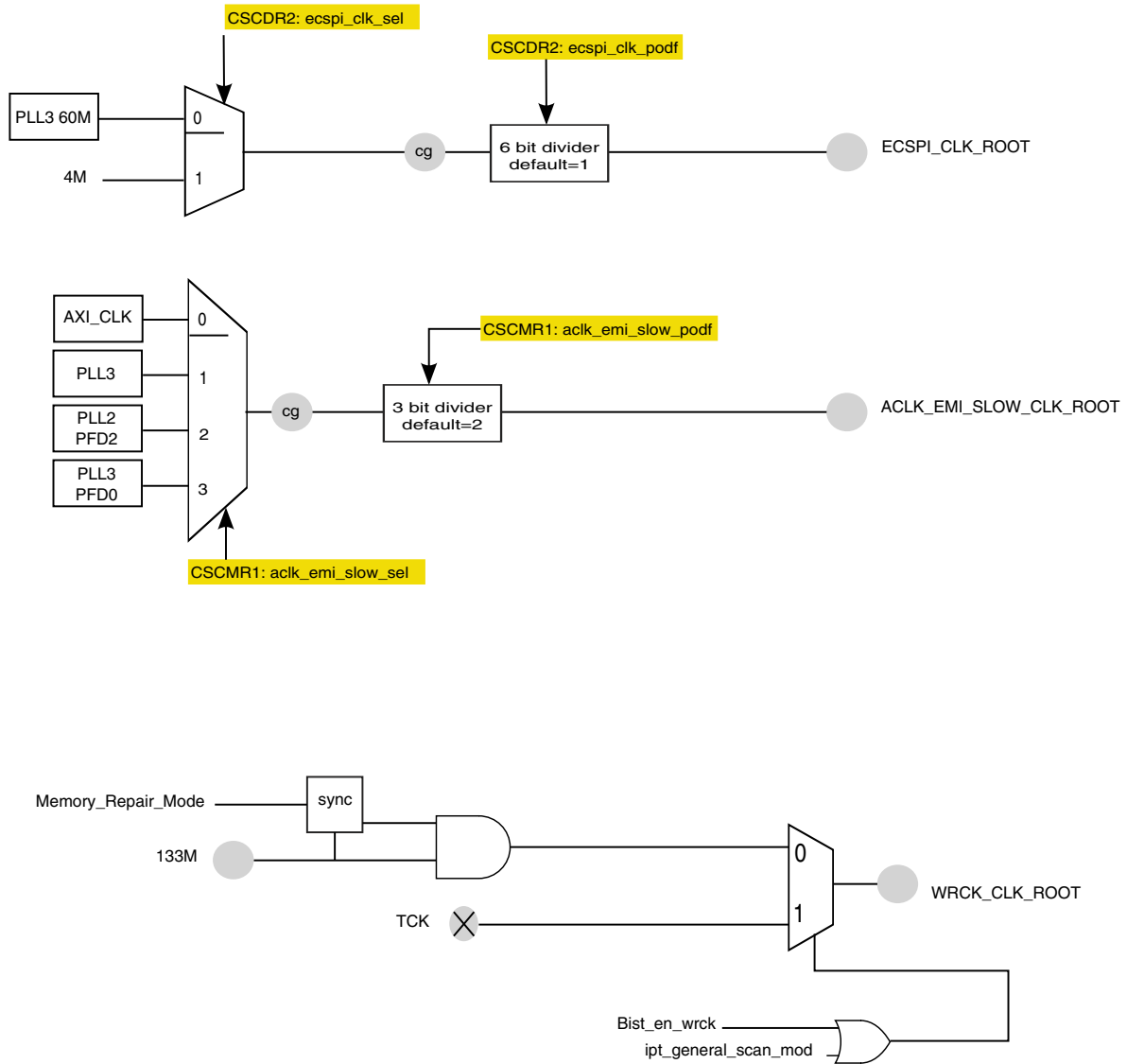


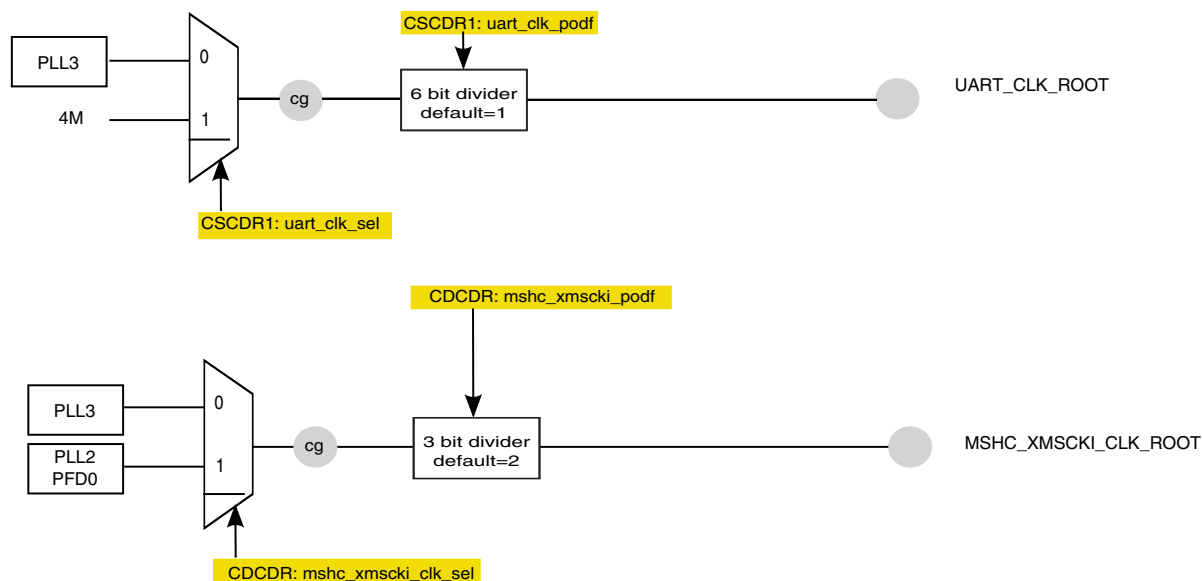




## Functional Description







**Figure 15-8. UART clock generation (cont)**

### NOTE

All 6-bit PODF dividers found in the diagrams above can operate on low frequency.

#### 15.5.1.5.5 Initial values controlled by the System JTAG Controller (SJC).

The initial values of the following dividers and muxes can be controlled by SJC.

In regular functional mode, the SJC will drive the reset values stated in the CCM register memory map. If SJC is programmed to change those values, then the reset value for those dividers/muxes will be taken from the SJC programmability.

Software can update the changed reset value after reset sequence. The control signals and the dividers/muxes are listed below:

- [2:0] init\_mmdc\_axi\_podf
- [2:0] init\_periph2\_clk2\_podf
- [1:0] init\_ipg\_podf
- [2:0] init\_ahb\_podf
- [2:0] init\_axi\_podf
- [2:0] init\_periph\_clk2\_podf
- init\_periph\_clk\_sel
- init\_periph2\_clk\_sel



#### 15.5.1.5.6 Divider change handshake

Modifying the following dividers will start the handshake with MMDC ROOT and/or CH0.

- mmdc\_ch0\_axi\_podf
- mmdc\_root\_axi\_podf
- periph\_clk\_sel
- periph2\_clk\_sel

#### 15.5.1.6 Disabling / Enabling PLLs

PLL disabling and enabling is done via analog module.

Before disabling a PLL using the analog registers, software should first move all the clocks generated from that specific PLL to another source. This alternate source could be another PLL, or a PFD driven by another PLL. Alternatively, software can bypass the PLL and use the PLL reference clock (usually 24MHz) as the output clock. Bypassing the PLL is done by setting the analog BYPASS bit in the control register for that PLL. For critical system bus clocks, changing the clock source can be done in the CCM using the glitchless clock muxes in [Figure 15-5](#). In the figure, the thick bar on the input side indicates the glitchless muxes. Those without the thick bar are regular muxes (not glitchless).

For example, before disabling PLL2, software can switch the FABRIC\_CLK\_ROOT away from the PLL2 or one of its PFDs by programming CBCMR[PERIPH2\_CLK2\_SEL] and CBCDR[PERIPH2\_CLK2\_PODF] to provide an appropriate frequency clock, then glitchlessly switch to it by programming CBCDR[PERIPH2\_CLK\_SEL].

For serial clocks, software should first disable the module, then gate its clock in the LPCG. Then it should move the mux controlling the source of the clocks to another PLL, and reset the module and its clocks. Only then is it safe to disable the PLL. The mux for the serial clocks is not glitchless.

#### 15.5.1.7 Low Power Clock Gating module (LPCG)

The LPCG module receives the root clocks and splits them to clock branches for each module. The clock branches are gated clocks.

The enables for those gates can come from four sources:

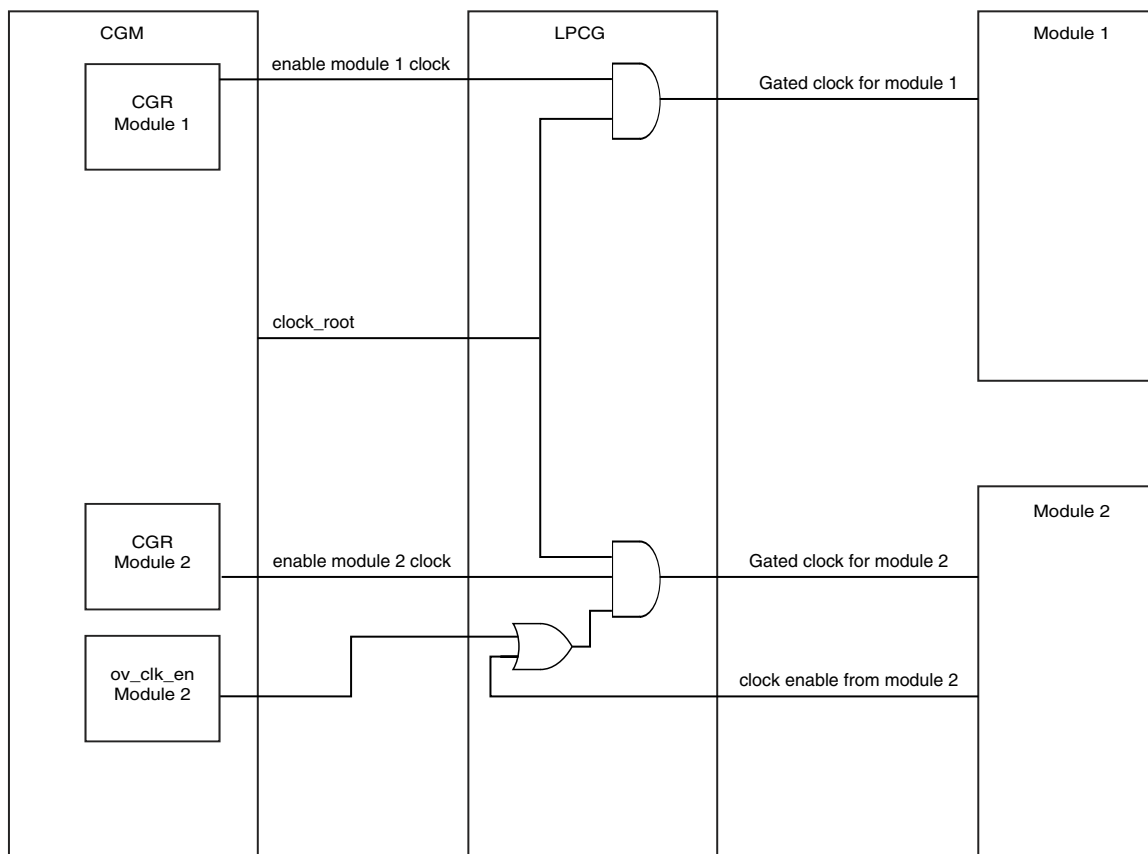
1. Clock enable signal from CCM - this signal is generated by configuration of the CGR bits in CCM. It is based on the low power mode.
2. Clock enable signal from the module - this signal is generated by the module based on internal logic of the module. Not every enable signal from the module is used. For used clock enable signals from the module, CCM will generate an override signal based on a programable bit in CCM (CMEOR).
3. Clock enable signal from Reset controller (SRC) - this signal will enable the clock during the reset procedure. Please see the SRC spec for details on the clock enable signal during reset procedure.
4. Hard-coded enable from fuse box.

These enable signals are ANDed to generate the enable signal for the gating cell.

The enable signal for the gating cell is synchronized with the clock it needs to gate in order to prevent glitches on the gated clock.

Notifications are generated for CCM to indicate when clock roots should be opened and closed. All notifications that correspond to the same clock root will be ORed to generate one notification signal to CCM for clock root gating.

The following figure describes the clock split inside the LPCG module. It describes the case of two modules; one module is without an enable signal and one is shown with an enable signal. SRC enable signals and sync flip flops are omitted from this figure.



**Figure 15-9. Clock split in LPCG**

#### 15.5.1.7.1 MMDC handshake

CCM will assert the `mmdc_freq_change_req` signal.

MMDC will assert the `mmdc0_freq_change_ack` and `mmdc1_freq_change_ack` signals to acknowledge that the frequency change request has been received and that the frequency can now be changed safely.

CCM will commence the actual change of division ratio of `mmdc0` and/or `mmdc1` dividers or apply mux change on root clocks once both of the non-masked acknowledges are asserted.

Note that `mmdc0` is not used on the i.MX 6SoloLite. Therefore the user should set register `CCDR[17]` to 1 before beginning any operation that initiates a handshake. It is acceptable to program and leave this override bit asserted.

## 15.5.2 DVFS support

When performing DVFS, the frequency shift procedure for the ARM core clock domain can be performed by software.

CPU PLL frequency or CCM ARM clk divider and CPU power domain supply voltage value are controlled by the CCM\_ANALOG module. Please note that frequency should be shifted down first and then voltage value reduced, and vice-versa, when shifting the frequency up.

Frequency shift for MMDC requires hardware support.

## 15.5.3 Power modes

The chip supports 3 low power modes: RUN mode, WAIT mode, STOP mode.

### 15.5.3.1 RUN mode

This is the normal/functional operating mode. In this mode, the CPU runs in its normal operational mode. Clocks to the modules can be gated by configuring the corresponding CCGRx bits.

### 15.5.3.2 WAIT mode

In this mode the CPU clock is gated. All other clocks are functional and can be gated by programming their CGR bits.

#### 15.5.3.2.1 Entering WAIT mode

If the CLPCR[LPM] bit is set by software to WAIT mode, when CPU executes the next wait for interrupt (WFI) instruction, WAIT mode sequence will start.

As part of the WFI routine, alternative interrupt controller in GPC should be updated; the CPU platform interrupt controller will be disabled first by software and will be not functional, due to clock gating. Interrupts during WAIT mode are monitored by alternative interrupt controller.

After execution of the WFI routine, the CPU platform will assert idle signals for each component of the platform and CCM will gate clock to the platform. Gating of ARM clock will take place only if CLPCR[5]=1 and debug\_arm\_clk\_off\_on\_lpm=0 (connected to the SJC register sjc\_gpucr3\_reg bit 15). If either CLPCR[5]=0 or debug\_arm\_clk\_off\_on\_lpm=1, the ARM clocks will not be gated off and CCM will continue to the next step.

The next actions can be programmed during WAIT mode:

1. CCM requests an acknowledge to close clocks to MMDC if its CGR bits indicate to close its clocks on WAIT mode, and if those clocks are not already closed in run mode. The request will be issued if the handshake is not bypassed by programming the CLPCR register. If the corresponding bits are set, the request signal will not be issued to the corresponding module and CCM will not wait for its acknowledge in the process of entering low power mode. Once CCM receives all the acknowledge signals needed, then it will enter WAIT mode.
2. Close the clocks to the modules which were defined to be shut at WAIT mode in the CCGR bits.
3. Observability to indicate WAIT mode.

Any enabled interrupt assertion will start the exit from WAIT mode.

### 15.5.3.2.2 Exiting WAIT mode

As soon as enabled interrupt is asserted, CPU supply will be restored if CPU SRPG was applied and clocks are enabled to CPU and other modules.

### 15.5.3.3 STOP mode

In this mode all system clocks are stopped, along with the CPU, system buses and all PLLs. Power gating is applied for ARM platform, GPU3D, GPU2D and VPU. External supply voltage can be reduced to decrease leakage.

#### 15.5.3.3.1 Entering STOP mode

Procedure entering STOP mode is the same, as entering WAIT mode until the moment of disabling clocks to modules. (LPM bit should be configured to STOP mode.)

After clocks to modules are gated, the following actions will be taken:

- PLLs are disabled
- pmic\_vstby\_req asserted, if vstby bit is set

- osc\_en signal is negated
- osc\_pwrdsn is asserted, if sbyos bit is set

Counter will be triggered after pmic\_vstby\_req assertion to allow to external regulator or PMIC to decrease voltage until valid voltage range. On counter completion, anatop\_stop\_mode signal will be asserted, that will trigger disabling analog elements in anatop.

CCM's low power state machine will remain in state STOP\_GPC until STOP mode is exited.

### **15.5.3.3.2 Exiting STOP mode**

As soon as an enabled interrupt is asserted, the CCM will begin the process of exiting STOP mode.

The following will take place:

1. If vstby bit was set, deassert PMIC\_STBY\_REQ to notify power management IC to change voltage from standby voltage to functional voltage.
2. If well bias was enabled, deasserted well bias controls.
3. If sbyos was set, and CCM closed either external oscillator or on board oscillator, then CCM will start oscillator by asserting ref\_en\_b signal and deasserting cosc\_pwrdown signal respectively.
4. After the amount of CKILs defined in stby\_count bits, wait until PMIC functional voltage is ready. This is the notification from power management IC that the voltage is ready at its functional value. Only then will CCM continue the steps.
5. Start osc. If oscillator was started, wait until oscnt has finished its counting to make sure that oscillator is ready.
6. Start PLLs. Only the PLLs that were configured to be on prior to the entrance to STOP mode will be started.
7. CCM will request GPC to restore ARM power by GPC\_PUP\_REQ. If power was removed from the ARM platform, GPC will notify CCM by asserting signal GPC\_PUP\_ACK that power to ARM is back on, and its safe to exit from STOP mode. Only then will the CCM progress to the next step.
8. Once assertion of notification from src that the resets for the power gated modules has been finished, (src\_power\_gating\_reset\_done is set) negate the low power request signals to all modules and enable all module clocks including ARM clocks and CKIL sync, and return to run mode. (Clocks whose CCGR bits are not to be opened in RUN mode will not be opened; they will continued to be gated.)

Once the system is in run mode, negate signals ccm\_ipg\_stop and system\_in\_stop\_mode.

## 15.6 CCM Memory Map/Register Definition

### NOTE

CCM Register reset values may not be the same in ROM. See for more information.

**CCM memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20C_4000	CCM Control Register (CCM_CCR)	32	R/W	0401_16FFh	<a href="#">15.6.1/524</a>
20C_4004	CCM Control Divider Register (CCM_CCDR)	32	R/W	0000_0000h	<a href="#">15.6.2/526</a>
20C_4008	CCM Status Register (CCM_CSR)	32	R	0000_0010h	<a href="#">15.6.3/527</a>
20C_400C	CCM Clock Swither Register (CCM_CCSR)	32	R/W	0000_0100h	<a href="#">15.6.4/528</a>
20C_4010	CCM Arm Clock Root Register (CCM_CACRR)	32	R/W	0000_0000h	<a href="#">15.6.5/530</a>
20C_4014	CCM Bus Clock Divider Register (CCM_CBCDR)	32	R/W	0001_8D00h	<a href="#">15.6.6/531</a>
20C_4018	CCM Bus Clock Multiplexer Register (CCM_CBCMR)	32	R/W	0082_0324h	<a href="#">15.6.7/533</a>
20C_401C	CCM Serial Clock Multiplexer Register 1 (CCM_CSCMR1)	32	R/W	00F0_0000h	<a href="#">15.6.8/536</a>
20C_4020	CCM Serial Clock Multiplexer Register 2 (CCM_CSCMR2)	32	R/W	02B9_2F06h	<a href="#">15.6.9/539</a>
20C_4024	CCM Serial Clock Divider Register 1 (CCM_CSCDR1)	32	R/W	0049_0B00h	<a href="#">15.6.10/539</a>
20C_4028	CCM SSI1 Clock Divider Register (CCM_CS1CDR)	32	R/W	0EC1_02C1h	<a href="#">15.6.11/542</a>
20C_402C	CCM SSI2 Clock Divider Register (CCM_CS2CDR)	32	R/W	0007_36C1h	<a href="#">15.6.12/544</a>
20C_4030	CCM D1 Clock Divider Register (CCM_CDCDR)	32	R/W	33F7_1F92h	<a href="#">15.6.13/545</a>
20C_4034	CCM HSC Clock Divider Register (CCM_CHSCCDR)	32	R/W	0002_A150h	<a href="#">15.6.14/547</a>
20C_4038	CCM Serial Clock Divider Register 2 (CCM_CSCDR2)	32	R/W	0002_9B48h	<a href="#">15.6.15/548</a>
20C_403C	CCM Serial Clock Divider Register 3 (CCM_CSCDR3)	32	R/W	0001_4841h	<a href="#">15.6.16/550</a>
20C_4048	CCM Divider Handshake In-Process Register (CCM_CDHIPR)	32	R	0000_0000h	<a href="#">15.6.17/552</a>
20C_4054	CCM Low Power Control Register (CCM_CLPCR)	32	R/W	0000_0079h	<a href="#">15.6.18/555</a>
20C_4058	CCM Interrupt Status Register (CCM_CISR)	32	w1c	0000_0000h	<a href="#">15.6.19/558</a>
20C_405C	CCM Interrupt Mask Register (CCM_CIMR)	32	R/W	FFFF_FFFFh	<a href="#">15.6.20/561</a>

*Table continues on the next page...*

## CCM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20C_4060	CCM Clock Output Source Register (CCM_CCOSR)	32	R/W	000A_0001h	<a href="#">15.6.21/564</a>
20C_4064	CCM General Purpose Register (CCM.CGPR)	32	R/W	0000_FE62h	<a href="#">15.6.22/566</a>
20C_4068	CCM Clock Gating Register 0 (CCM_CCGR0)	32	R/W	FFFF_FFFFh	<a href="#">15.6.23/567</a>
20C_406C	CCM Clock Gating Register 1 (CCM_CCGR1)	32	R/W	FFFF_FFFFh	<a href="#">15.6.24/569</a>
20C_4070	CCM Clock Gating Register 2 (CCM_CCGR2)	32	R/W	FC3F_FFFFh	<a href="#">15.6.25/570</a>
20C_4074	CCM Clock Gating Register 3 (CCM_CCGR3)	32	R/W	FFFF_FFFFh	<a href="#">15.6.26/572</a>
20C_4078	CCM Clock Gating Register 4 (CCM_CCGR4)	32	R/W	FFFF_FFFFh	<a href="#">15.6.27/573</a>
20C_407C	CCM Clock Gating Register 5 (CCM_CCGR5)	32	R/W	FFFF_FFFFh	<a href="#">15.6.28/574</a>
20C_4080	CCM Clock Gating Register 6 (CCM_CCGR6)	32	R/W	FFFF_FFFFh	<a href="#">15.6.29/576</a>
20C_4088	CCM Module Enable Override Register (CCM_CMEOR)	32	R/W	FFFF_FFFFh	<a href="#">15.6.30/577</a>

## 15.6.1 CCM Control Register (CCM\_CCR)

The figure below represents the CCM Control Register (CCR), which contains bits to control general operation of CCM. The table below provides its field descriptions.

Address: 20C\_4000h base + 0h offset = 20C\_4000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				RBC_EN	REG_BYPASS_COUNT						0	WB_COUNT			
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			COSC_EN	0				OSCNT							
W																
Reset	0	0	0	1	0	1	1	0	1	1	1	1	1	1	1	1



## CCM\_CCR field descriptions

Field	Description
31–28 Reserved	This read-only field is reserved and always has the value 0.
27 RBC_EN	Enable for REG_BYPASS_COUNTER. If enabled, analog_reg_bypass signal will be asserted after REG_BYPASS_COUNT clocks of CKIL, after standby voltage is requested. If standby voltage is not requested analog_reg_bypass won't be asserted, event if counter is enabled.  1 REG_BYPASS_COUNTER enabled. 0 REG_BYPASS_COUNTER disabled
26–21 REG_BYPASS_COUNT	Counter for analog_reg_bypass signal assertion after standby voltage request by pmic_vstby_req. Should be zeroed and reconfigured after exit from low power mode.  REG_BYPASS_COUNT can also be used for holding off interrupts when the PGC unit is sending signals to power gate the core.  000000 no delay 000001 1 CKIL clock period delay 111111 63 CKIL clock periods delay
20–19 Reserved	This read-only field is reserved and always has the value 0.
18–16 WB_COUNT	Well Bias counter. Delay, defined by this value, counted by CKIL clock will be applied till well bias is enabled at exit from wait or stop low power mode. Counter will be used if wb_core_at_lpm or wb_per_at_lpm bits are set. Should be zeroed and reconfigured after exit from low power mode.  000 no delay 001 1 CKIL clock delay 111 7 CKIL clocks delay
15–13 Reserved	This read-only field is reserved and always has the value 0.
12 COSC_EN	On chip oscillator enable bit - this bit value is reflected on the output cosc_en. The system will start with on chip oscillator enabled to supply source for the PLLs. Software can change this bit if a transition to the bypass PLL clocks was performed for all the PLLs. In cases that this bit is changed from '0' to '1' then CCM will enable the on chip oscillator and after counting oscnt ckil clock cycles it will notify that on chip oscillator is ready by a interrupt cosc_ready and by status bit cosc_ready. The cosc_en bit should be changed only when on chip oscillator is not chosen as the clock source.  0 disable on chip oscillator 1 enable on chip oscillator
11–8 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
7–0 OSCNT	Oscillator ready counter value. These bits define value of 32KHz counter, that serve as counter for oscillator lock time. This is used for oscillator lock time. Current estimation is ~5ms. This counter will be used in ignition sequence and in wake from stop sequence if sbyos bit was defined, to notify that on chip oscillator output is ready for the dp1l_ip to use and only then the gate in dp1l_ip can be opened.  00000000 count 1 ckil 11111111 count 256 ckil's (default)

## 15.6.2 CCM Control Divider Register (CCM\_CCDDR)

The figure below represents the CCM Control Divider Register (CCDR), which contains bits that control the loading of the dividers that need handshake with the modules they affect. The table below provides its field descriptions.

Address: 20C\_4000h base + 4h offset = 20C\_4004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0														mmdc_ch0_mask	mmdc_root_mask
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

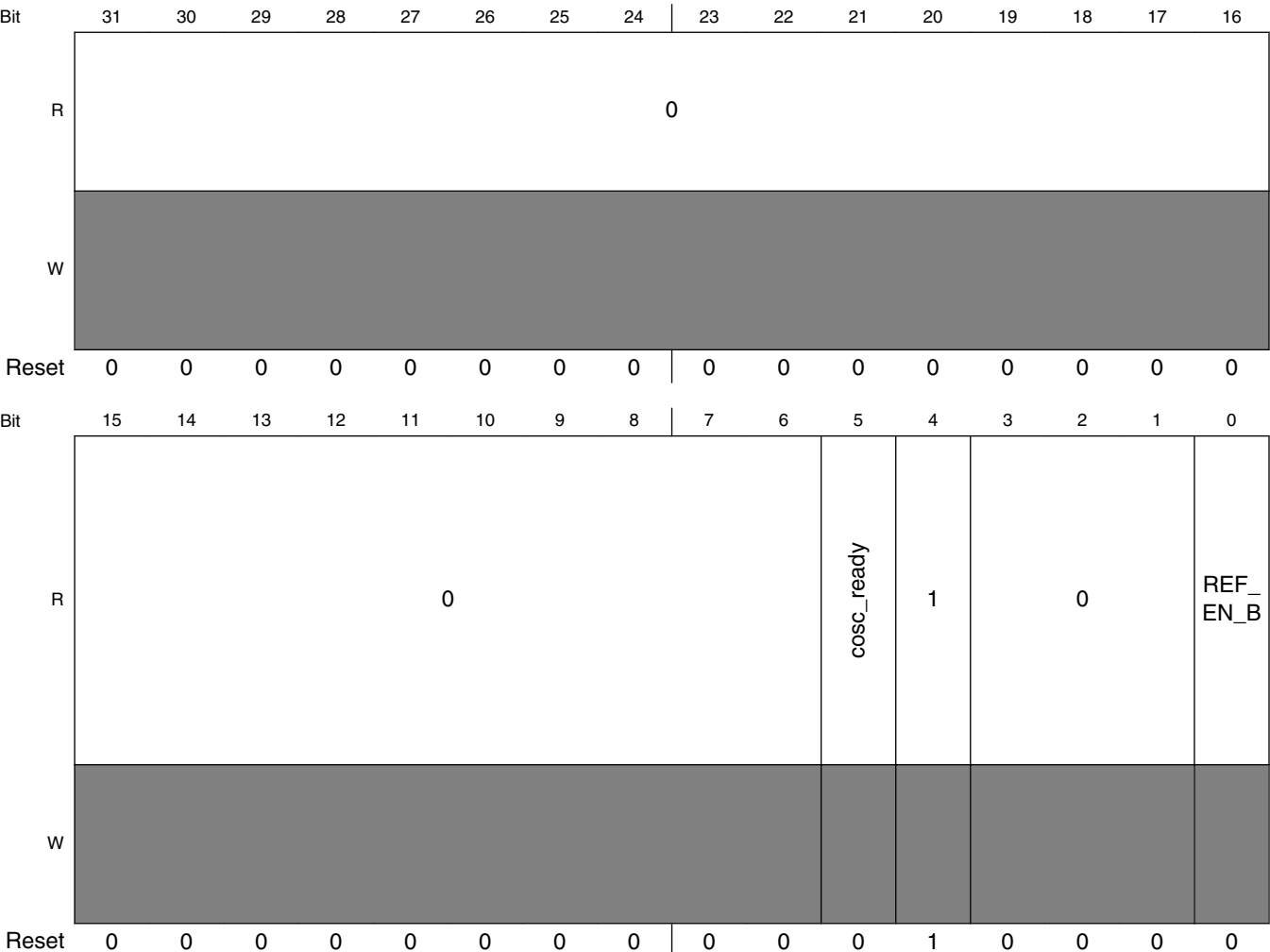
### CCM\_CCDDR field descriptions

Field	Description
31–18 Reserved	This read-only field is reserved and always has the value 0.
17 mmdc_ch0_mask	During divider ratio mmdc_ch0_axi_podf change or sync mux periph_clk_sel change (but not jtag) or SRC request during warm reset, mask handshake with mmdc_ch0 module.  0 allow handshake with mmdc_ch0 module 1 mask handshake with mmdc_ch0. Request signal will not be generated.
16 mmdc_root_mask	During divider ratio mmdc_root_axi_podf change or sync mux periph2_clk_sel change (but not jtag) or SRC request during warm reset, mask handshake with mmdc_root module.  0 allow handshake with mmdc_root module 1 mask handshake with mmdc_root. Request signal will not be generated.
15–0 Reserved	This read-only field is reserved and always has the value 0.

### 15.6.3 CCM Status Register (CCM\_CSR)

The figure below represents the CCM status Register (CSR). The status bits are read-only bits. The table below provides its field descriptions.

Address: 20C\_4000h base + 8h offset = 20C\_4008h



CCM\_CSR field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value 0.
5 cosc_ready	Status indication of on board oscillator. This bit will be asserted if on chip oscillator is enabled and on chip oscillator is not powered down, and if oscnt counter has finished counting.

Table continues on the next page...

**CCM\_CSR field descriptions (continued)**

Field	Description
	0 on board oscillator is not ready. 1 on board oscillator is ready.
4 Reserved	This read-only field is reserved and always has the value 1.
3–1 Reserved	This read-only field is reserved and always has the value 0.
0 REF_EN_B	Status of the value of CCM_REF_EN_B output of ccm 0 value of CCM_REF_EN_B is '0' 1 value of CCM_REF_EN_B is '1'

**15.6.4 CCM Clock Swither Register (CCM\_CCSR)**

The figure below represents the CCM Clock Switcher register (CCSR). The CCSR register contains bits to control the switcher sub-module dividers and multiplexers. The table below provides its field descriptions.

Address: 20C\_4000h base + Ch offset = 20C\_400Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	pfd_540m_dis_mask	pfd_720m_dis_mask	pfd_454m_dis_mask	pfd_508m_dis_mask	pfd_594m_dis_mask	pfd_352m_dis_mask	pfd_396m_dis_mask	step_sel	0							
W													pll1_sw_clk_sel	pll2_sw_clk_sel	pll3_sw_clk_sel	
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

**CCM\_CCSR field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**CCM\_CCSR field descriptions (continued)**

Field	Description
15 pfd_540m_dis_mask	Mask of 540M PFD auto-disable. 0 - 540M PFD disable=0 (PFD always on) 1 540M PFD disable is managed by associated dividers disable. If all 540M-driven dividers are closed, PFD is disabled.
14 pfd_720m_dis_mask	Mask of 720M PFD auto-disable. 0 720M PFD disable=0 (PFD always on) 1 720M PFD disable is managed by associated dividers disable. If all 720M-driven dividers are closed, PFD is disabled.
13 pfd_454m_dis_mask	Mask of 454M PFD auto-disable. 0 454M PFD disable=0 (PFD always on) 1 454M PFD disable is managed by associated dividers disable. If all 454M-driven dividers are closed, PFD is disabled.
12 pfd_508m_dis_mask	Mask of 508M PFD auto-disable. 0 508M PFD disable=0 (PFD always on) 1 508M PFD disable is managed by associated dividers disable. If all 508M-driven dividers are closed, PFD is disabled.
11 pfd_594m_dis_mask	Mask of 594M PFD auto-disable. 0 594M PFD disable=0 (PFD always on) 1 594M PFD disable is managed by associated dividers disable. If all 594M-driven dividers are closed, PFD is disabled.
10 pfd_352m_dis_mask	Mask of 352M PFD auto-disable. 0 352M PFD disable=0 (PFD always on) 1 352M PFD disable is managed by associated dividers disable. If all 352M-driven dividers are closed, PFD is disabled.
9 pfd_396m_dis_mask	Mask of 396M PFD auto-disable. 0 396M PFD disable=0 (PFD always on) 1 396M PFD disable is managed by associated dividers disable. If all 396M-driven dividers are closed, PFD is disabled.
8 step_sel	Selects the option to be chosen for the step frequency when shifting ARM frequency. This will control the step_clk.  <b>NOTE:</b> This mux is allowed to be changed only if its output is not used, i.e. ARM uses the output of pll1, and step_clk is not used.  0 osc_clk (24M) - source for lp_apm. (default) 1 pll2 PFD clock
7-3 Reserved	This read-only field is reserved and always has the value 0.
2 pll1_sw_clk_sel	Selects source to generate pll1_sw_clk. 0 pll1_main_clk(default) 1 step_clk

*Table continues on the next page...*

## CCM\_CCSR field descriptions (continued)

Field	Description
1 pll2_sw_clk_sel	Selects source to generate pll2_sw_clk. This bit should only be used for testing purposes. 0 pll2_main_clk(default) 1 pll2 bypass clock
0 pll3_sw_clk_sel	Selects source to generate pll3_sw_clk. This bit should only be used for testing purposes. 0 pll3_main_clk(default) 1 pll3 bypass clock

## 15.6.5 CCM Arm Clock Root Register (CCM\_CACRR)

The figure below represents the CCM Arm Clock Root register (CACRR). The CACRR register contains bits to control the ARM clock root generation. The table below provides its field descriptions.

Address: 20C\_4000h base + 10h offset = 20C\_4010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																	0															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## CCM\_CACRR field descriptions

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value 0.
2–0 arm_podf	<p>Divider for ARM clock root.</p> <p><b>NOTE:</b> If arm_freq_shift_divider is set to '1' then any new write to arm_podf will be held until arm_clk_switch_req signal is asserted.</p> <p>000 divide by 1(default) 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8</p>

## 15.6.6 CCM Bus Clock Divider Register (CCM\_CBCDR)

The figure below represents the CCM Bus Clock Divider Register (CBCDR). The CBCDR register contains bits to control the clock generation sub module dividers. The table below provides its field descriptions.

Address: 20C\_4000h base + 14h offset = 20C\_4014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		periph_clk2_podf				periph2_clk_sel	periph_clk_sel	Reserved						ocram_podf	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved			ahb_podf				ipg_podf		ocram_alt_clk_sel	ocram_clk_sel	fabric_mmdc_podf		periph2_clk2_podf		
W																
Reset	1	0	0	0	1	1	0	1	0	0	0	0	0	0	0	0

**CCM\_CBCDR field descriptions**

Field	Description
31–30 -	This field is reserved. Reserved
29–27 periph_clk2_podf	Divider for periph2 clock podf.  000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
26 periph2_clk_sel	Selector for peripheral2 main clock (source of mmdc_root_axi_clk_root ).  0 derive clock from pll2_sw_clk muxed clock source. 1 derive clock from periph_clk2_clk clock source.
25 periph_clk_sel	Selector for peripheral main clock (source of mmdc_ch0_axi_clk_root).

*Table continues on the next page...*

## CCM\_CBCDR field descriptions (continued)

Field	Description
	<p><b>NOTE:</b> Alternative clock source should be used when PLL is relocked. For PLL relock procedure pls refer PLL chapter.</p> <p>0    derive clock from pll2_sw_clk muxed clock source.</p> <p>1    derive clock from periph_clk2_clk clock source.</p>
24–19 -	This field is reserved. Reserved
18–16 ocram_podf	<p>Post divider for ocram clock.</p> <p><b>NOTE:</b> Any change of this divider might involve handshake with EMI. See CDHIPR register for the handshake busy bits.</p> <p>000    divide by 1</p> <p>001    divide by 2</p> <p>010    divide by 3</p> <p>011    divide by 4</p> <p>100    divide by 5</p> <p>101    divide by 6</p> <p>110    divide by 7</p> <p>111    divide by 8</p>
15–13 -	This field is reserved. Reserved
12–10 ahb_podf	<p>Divider for AHB PODF.</p> <p><b>NOTE:</b> Any change of this divider might involve handshake with EMI. See CDHIPR register for the handshake busy bits.</p> <p>000    divide by 1</p> <p>001    divide by 2</p> <p>010    divide by 3</p> <p>011    divide by 4</p> <p>100    divide by 5</p> <p>101    divide by 6</p> <p>110    divide by 7</p> <p>111    divide by 8</p>
9–8 ipg_podf	<p>Divider for ipg podf.</p> <p><b>NOTE:</b> IEEE_RTC module will not support ratio of 1:3 for ahb_clk:ipg_clk. In case IEEE_RTC is used, then those ratios should not be used.</p> <p><b>NOTE:</b> SDMA module will not support ratio of 1:3 and 1:4 for ahb_clk:ipg_clk. In case SDMA is used, then those ratios should not be used.</p> <p>00    divide by 1</p> <p>01    divide by 2</p> <p>10    divide by 3</p> <p>11    divide by 4</p>
7 ocram_alt_clk_sel	<p>OCRAM alternative clock select</p> <p>0    pll2 396MHz PFD will be selected as alternative clock for OCRAM root clock</p> <p>1    pll3 540MHz PFD will be selected as alternative clock for OCRAM root clock</p>

Table continues on the next page...



**CCM\_CBCDR field descriptions (continued)**

Field	Description
6 ocram_clk_sel	OCRAM clock source select 0 Periph_clk output will be used as OCRAM clock root 1 AXI alternative clock will be used as OCRAM clock root
5–3 fabric_mmdc_podf	Post divider for fabric_mmdc clock. <b>NOTE:</b> This design implementation does not use MMDC_ROOT_AXI_CLK_ROOT as a clock source to the MMDC. Only MMDC_CH0_AXI_CLK_ROOT is used.  000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
2–0 periph2_clk2_podf	Divider for periph2_clk2 podf. <b>NOTE:</b> Divider should be updated when output clock is gated.  000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8

**15.6.7 CCM Bus Clock Multiplexer Register (CCM\_CBCMR)**

The figure below represents the CCM Bus Clock Multiplexer Register (CBCMR). The CBCMR register contains bits to control the multiplexers that generate the bus clocks. The table below provides its field descriptions.

**NOTE**

Any change on the above multiplexer will have to be done while the module that its clock is affected is not functional and the respective clock is gated in LPCG. If the change will be done during operation of the module, then it is not guaranteed that the modules operation will not be harmed.

The change for arm\_clk\_sel should be done through sdma so that ARM will not use this clock during the change and the clock will be gated in LPCG.

## CCM Memory Map/Register Definition

Address: 20C\_4000h base + 18h offset = 20C\_4018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	gpu2d_core_podf			gpu2d_ovg_core_podf			epdc_pix_podf			pre_periph2_clk_sel		periph2_clk2_sel	pre_periph_clk_sel		Reserved	
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		periph_clk2_sel		vdoaxi_clk_sel	Reserved	gpu2d_core_clk_sel		Reserved		gpu2d_ovg_core_clk_sel	Reserved			Reserved	Reserved
W																
Reset	0	0	0	0	0	0	1	1	0	0	1	0	0	1	0	0

### CCM\_CBCMR field descriptions

Field	Description
31–29 gpu2d_core_podf	<p>Post divider for gpu2d_core clock.</p> <p><b>NOTE:</b> Divider should be updated when output clock is gated.</p> <p>000 divide by 1            001 divide by 2            010 divide by 3            011 divide by 4            100 divide by 5            101 divide by 6            110 divide by 7            111 divide by 8</p>
28–26 gpu2d_ovg_core_podf	<p>Divider for gpu3d_core clock.</p> <p><b>NOTE:</b> Divider should be updated when output clock is gated.</p> <p>000 divide by 1            001 divide by 2            010 divide by 3            011 divide by 4            100 divide by 5            101 divide by 6            110 divide by 7            111 divide by 8</p>

Table continues on the next page...

## CCM\_CBCMR field descriptions (continued)

Field	Description
25–23 epdc_pix_podf	Post divider for EPDC_PIX.  <b>NOTE:</b> Divider should be updated when output clock is gated.  000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
22–21 pre_periph2_clk_sel	Selector for pre_periph2 clock multiplexer  00 derive clock from PLL2 main 528MHz clock 01 derive clock from 396MHz PLL2 PFD 10 derive clock from 352M PFD 11 derive clock from 198MHz clock (divided 396MHz PLL2 PFD)
20 periph2_clk2_sel	Selector for periph2_clk2 clock multiplexer  0 derive clock from pll3_sw_clk 1 derive clock from PLL2 Main
19–18 pre_periph_clk_sel	Selector for pre_periph clock multiplexer  00 derive clock from PLL2 main 528MHz clock 01 derive clock from 396MHz PLL2 PFD 10 derive clock from 352M PFD 11 derive clock from 198MHz clock (divided 396MHz PLL2 PFD)
17–16 -	This field is reserved. Reserved
15–14 -	This field is reserved. Reserved
13–12 periph_clk2_sel	Selector for peripheral clk2 clock multiplexer  00 derive clock from pll3_sw_clk 01 derive clock from pll1_ref_clk 10 derive clock from pll2_burn_in_clk 11 reserved
11 vdoaxi_clk_sel	Selector for vdoaxi clock multiplexer  0 derive clock from axi clk 1 derive clock from 132M clock
10 -	This field is reserved. Reserved
9–8 gpu2d_core_clk_sel	Selector for gpu2d_core clock multiplexer  00 derive clock from mmdc_ch0 clk 01 derive clock from pll3

*Table continues on the next page...*

**CCM\_CBCMR field descriptions (continued)**

Field	Description
10	derive clock from 594M PFD
11	derive clock from 720M PFD
7–6 -	This field is reserved. Reserved
5–4 gpu2d_ovg_ core_clk_sel	Selector for gpu2d_ovg_core clock multiplexer
3–2 -	This field is reserved. Reserved
1 -	This field is reserved. Reserved
0 -	This field is reserved. Reserved

**15.6.8 CCM Serial Clock Multiplexer Register 1 (CCM\_CSCMR1)**

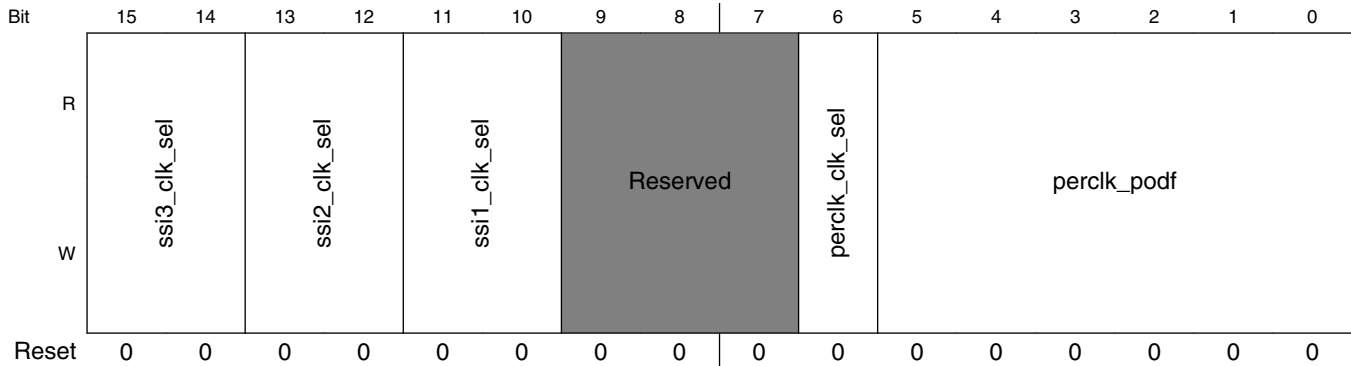
The figure below represents the CCM Serial Clock Multiplexer Register 1 (CSCMR1). The CSCMR1 register contains bits to control the multiplexers that generate the serial clocks. The table below provides its field descriptions.

**NOTE**

Any change on the above multiplexer will have to be done while the module that its clock is affected is not functional and the clock is gated. If the change will be done during operation of the module, then it is not guaranteed that the modules operation will not be harmed.

Address: 20C\_4000h base + 1Ch offset = 20C\_401Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0



CCM\_CSCMR1 field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–29 aclk_eim_slow_sel	Selector for aclk_eim_slow root clock multiplexer 00 derive clock from AXI clk root (default) 01 derive clock from PLL3 10 derive clock from 396M PFD 11 derive clock from 352M PFD
28–27 -	This field is reserved. Reserved
26 -	This field is reserved. Reserved
25–23 aclk_eim_slow_podf	Divider for aclk_eim_slow clock root. 000 divide by 1 001 divide by 2 (default) 111 divide by 8
22–20 lcdif_pix_podf	Post divider for LCDIF_PIX. <b>NOTE:</b> These bits are inverted between R/W and are not sequential. 000 divide by 7 (Read value 110) 001 divide by 8 (Read value 111) 010 divide by 5 (Read value 100) 011 divide by 6 (Read value 101) 100 divide by 3 (Read value 010) 101 divide by 4 (Read value 011) 110 divide by 1 (Read value 000) 111 divide by 2 (default) (Read value 001)
19 usdhc4_clk_sel	Selector for usdhc4 clock multiplexer 0 derive clock from 396M PFD 1 derive clock from 352M PFD
18 usdhc3_clk_sel	Selector for usdhc3 clock multiplexer

Table continues on the next page...

**CCM\_CSCMR1 field descriptions (continued)**

Field	Description
	0 derive clock from 396M PFD 1 derive clock from 352M PFD
17 usdhc2_clk_sel	Selector for usdhc2 clock multiplexer 0 derive clock from 396M PFD 1 derive clock from 352M PFD
16 usdhc1_clk_sel	Selector for usdhc1 clock multiplexer 0 derive clock from 396M PFD 1 derive clock from 352M PFD
15–14 ssi3_clk_sel	Selector for ssi3 clock multiplexer 00 derive clock from 508.2M PFD (default) 01 derive clock from 454.7M PFD 10 derive clock from pll4 11 Restricted
13–12 ssi2_clk_sel	Selector for ssi2 clock multiplexer 00 derive clock from 508.2M PFD (default) 01 derive clock from 454.7M PFD 10 derive clock from pll4 11 Restricted
11–10 ssi1_clk_sel	Selector for ssi1 clock multiplexer 00 derive clock from 508.2M PFD (default) 01 derive clock from 454.7M PFD 10 derive clock from pll4 11 Restricted
9–7 -	This field is reserved. Reserved
6 perclk_clk_sel	Selector for the perclk clock multiplexor 0 ipg clk root 1 OSC clock
5–0 perclk_podf	Divider for perclk podf. 000 divide by 1 (default) 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8

### 15.6.9 CCM Serial Clock Multiplexer Register 2 (CCM\_CSCMR2)

The figure below represents the CCM Serial Clock Multiplexer Register 2 (CSCMR2). The CSCMR2 register contains bits to control the multiplexers that generate the serial clocks. The table below provides its field descriptions.

#### NOTE

Any change on the above multiplexer will have to be done while the module that its clock is affected is not functional and the clock is gated. If the change will be done during operation of the module, then it is not guaranteed that the modules operation will not be harmed.

Address: 20C\_4000h base + 20h offset = 20C\_4020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved											extern_ audio_clk_sel	Reserved			
W																
Reset	0	0	0	0	0	0	1	0	1	0	1	1	1	0	0	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	1	0	1	1	1	1	0	0	0	0	0	1	1	0

#### CCM\_CSCMR2 field descriptions

Field	Description
31–21 -	This field is reserved. Reserved
20–19 extern_audio_ clk_sel	Selector for external audio clock multiplexer 00 derive clock from pll4 divided clock 01 derive clock from 508M PFD clock 10 derive clock from 454M PFD clock 11 derive clock from pll3 clock
18–0 -	This field is reserved. Reserved

### 15.6.10 CCM Serial Clock Divider Register 1 (CCM\_CSCDR1)

The figure below represents the CCM Serial Clock Divider Register 1 (CSCDR1). The CSCDR1 register contains bits to control the clock generation sub-module dividers. The table below provides its field descriptions.

**NOTE**

Any change on the above dividers will have to be done while the module that its clock is affected is not functional and the affected clock is gated. If the change will be done during operation of the module, then it is not guaranteed that the modules operation will not be harmed.

Address: 20C\_4000h base + 24h offset = 20C\_4024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				Reserved				usdhc4_podf			usdhc3_podf			usdhc2_podf	
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		usdhc1_podf			Reserved			uart_clk_sel		uart_clk_podf					
W																
Reset	0	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0

**CCM\_CSCDR1 field descriptions**

Field	Description
31–28 Reserved	This read-only field is reserved and always has the value 0.
27–25 -	This field is reserved. Reserved
24–22 usdhc4_podf	Divider for esdhc4 clock pred. <b>NOTE:</b> Divider should be updated when output clock is gated.  000 divide by 1 001 divide by 2 (default) 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
21–19 usdhc3_podf	Divider for usdhc3 clock podf. <b>NOTE:</b> Divider should be updated when output clock is gated.  000 divide by 1 001 divide by 2 (default) 010 divide by 3

Table continues on the next page...



**CCM\_CSCDR1 field descriptions (continued)**

Field	Description
	011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
18–16 usdhc2_podf	Divider for usdhc2 clock. <b>NOTE:</b> Divider should be updated when output clock is gated. 000 divide by 1 001 divide by 2 (default) 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
15–14 -	This field is reserved. Reserved
13–11 usdhc1_podf	Divider for usdhc1 clock podf. <b>NOTE:</b> Divider should be updated when output clock is gated. 000 divide by 1 001 divide by 2 (default) 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
10–7 -	This field is reserved. Reserved.
6 uart_clk_sel	Selector for the UART clock multiplexor 0 pll3_80m 1 OSC clk
5–0 uart_clk_podf	Divider for uart clock podf. 000000 divide by 1 (default) 111111 divide by 2 <sup>6</sup>

## 15.6.11 CCM SSI1 Clock Divider Register (CCM\_CS1CDR)

The figure below represents the CCM SSI1, SSI3, ESAI Clock Divider Register (CS1CDR). The CS1CDR register contains bits to control the ssi1 clock generation dividers. The table below provides its field descriptions.

Address: 20C\_4000h base + 28h offset = 20C\_4028h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				extern_audio_clk_podf				ssi3_clk_pred				ssi3_clk_podf			
W																
Reset	0	0	0	0	1	1	1	0	1	1	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				extern_audio_clk_pred				ssi1_clk_pred				ssi1_clk_podf			
W																
Reset	0	0	0	0	0	0	1	0	1	1	0	0	0	0	0	1

### CCM\_CS1CDR field descriptions

Field	Description
31–28 Reserved	This read-only field is reserved and always has the value 0.
27–25 extern_audio_clk_podf	Divider for external audio clock podf. 000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
24–22 ssi3_clk_pred	Divider for ssi3 clock pred. 000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
21–16 ssi3_clk_podf	Divider for ssi3 clock podf. The input clock to this divider should be lower than 300Mhz, the predivider can be used to achieve this.

*Table continues on the next page...*

**CCM\_CS1CDR field descriptions (continued)**

Field	Description
	000000 divide by 1 111111 divide by 2 <sup>6</sup>
15–12 Reserved	This read-only field is reserved and always has the value 0.
11–9 extern_audio_ clk_pred	Divider for external audio clock pred.  000 divide by 1 001 divide by 2 (default) 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
8–6 ssi1_clk_pred	Divider for ssi1 clock pred.  000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
5–0 ssi1_clk_podf	Divider for ssi1 clock podf. The input clock to this divider should be lower than 300Mhz, the predivider can be used to achieve this.  000000 divide by 1 111111 divide by 2 <sup>6</sup>

## 15.6.12 CCM SSI2 Clock Divider Register (CCM\_CS2CDR)

The figure below represents the CCM SSI2, LDB Clock Divider Register (CS2CDR). The CS2CDR register contains bits to control the ssi2 clock generation dividers, and ldb serial clocks select. The table below provides its field descriptions.

Address: 20C\_4000h base + 2Ch offset = 20C\_402Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	ldb_di1_clk_sel			ldb_di0_clk_sel			ssi2_clk_pred			ssi2_clk_podf					
W																
Reset	0	0	1	1	0	1	1	0	1	1	0	0	0	0	0	1

### CCM\_CS2CDR field descriptions

Field	Description
31–15 -	This field is reserved. Reserved
14–12 ldb_di1_clk_sel	Selector for ldb_di1 clock multiplexer 000 pll5 clock 001 pll2 352M PFD (default) 010 pll2 396M PFD 011 MMDC_ROOT clock 100 pll3 clock 101 111 Resrticted
11–9 ldb_di0_clk_sel	Selector for ldb_di1 clock multiplexer 000 pll5 clock 001 pll2 352M PFD (default) 010 pll2 396M PFD 011 MMDC_ROOT clock 100 pll3 clock 101 111 Resrticted
8–6 ssi2_clk_pred	Divider for ssi2 clock pred. 000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6

Table continues on the next page...

**CCM\_CS2CDR field descriptions (continued)**

Field	Description
	110 divide by 7 111 divide by 8
5–0 ssi2_clk_podf	Divider for ssi2 clock podf. The input clock to this divider should be lower than 300Mhz, the predivider can be used to achieve this.  000000 divide by 1 111111 divide by 2 <sup>6</sup>

**15.6.13 CCM D1 Clock Divider Register (CCM\_CDCDR)**

The figure below represents the CCM DI Clock Divider Register (CDCDR). The table below provides its field descriptions.

Address: 20C\_4000h base + 30h offset = 20C\_4030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
R	Reserved				spdif0_clk_pred				spdif0_clk_podf				spdif0_clk_sel		Reserved			
W	Reserved				spdif0_clk_pred				spdif0_clk_podf				spdif0_clk_sel		Reserved			
Reset	0	0	1	1	0	0	1	1	1	1	1	1	0	1	1	1		

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	spdif1_clk_pred				spdif1_clk_podf				spdif1_clk_sel		Reserved				
W	0	spdif1_clk_pred				spdif1_clk_podf				spdif1_clk_sel		Reserved				
Reset	0	0	0	1	1	1	1	1	1	0	0	1	0	0	1	0

**CCM\_CDCDR field descriptions**

Field	Description
31–28 -	This field is reserved. Reserved  0 derive from pll3 120M clock (default) 1 derive from pll2 396M PFD
27–25 spdif0_clk_pred	Divider for spdif0 clock pred. <b>NOTE:</b> Divider should be updated when output clock is gated.  000 divide by 1 (do not use with high input frequencies)

*Table continues on the next page...*

## CCM\_CDCDR field descriptions (continued)

Field	Description
	001 divide by 2 010 divide by 3 (default) 111 divide by 8
24–22 spdif0_clk_podf	Divider for spdif0 clock podf. <b>NOTE:</b> Divider should be updated when output clock is gated. 000 divide by 1 111 divide by 8
21–20 spdif0_clk_sel	Selector for spdif0 clock multiplexer 00 derive clock from pll4 divided clock 01 derive clock from 508M PFD clock 10 derive clock from 454M PFD clock 11 derive clock from pll3 clock
19–16 -	This field is reserved. Reserved
15 Reserved	This read-only field is reserved and always has the value 0.
14–12 spdif1_clk_pred	Divider for spdif1 clock pred. <b>NOTE:</b> Divider should be updated when output clock is gated. 000 divide by 1 (do not use with high input frequencies) 001 divide by 2 010 divide by 3 (default) 111 divide by 8
11–9 spdif1_clk_podf	Divider for spdif1 clock podf. <b>NOTE:</b> Divider should be updated when output clock is gated. 000 divide by 1 111 divide by 8
8–7 spdif1_clk_sel	Selector for spdif1 clock multiplexer 00 derive clock from pll4 divided clock 01 derive clock from 508M PFD clock 10 derive clock from 454M PFD clock 11 derive clock from pll3 clock
6–0 -	This field is reserved. Reserved

### 15.6.14 CCM HSC Clock Divider Register (CCM\_CHSCCCR)

The figure below represents the CCM HSC Clock Divider Register (CHSCCCR). The CHSCCCR register contains bits to control the EPDC (SiPix) and PXP clock generation dividers. The table below provides its field descriptions.

Address: 20C\_4000h base + 34h offset = 20C\_4034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														epdc_axi_clk_sel		epdc_axi_podf		Reserved			pxp_axi_clk_sel		pxp_axi_podf		Reserved						
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	0	0	0	1	0	1	0	1	0	0	0	0

**CCM\_CHSCCCR field descriptions**

Field	Description
31–18 Reserved	This read-only field is reserved and always has the value 0.
17–15 epdc_axi_clk_sel	Selector for epdc_axi (SiPix) root clock multiplexer 000      derive clock from mmdc_ch0 clock 001      derive clock from pll3 010      derive clock from pll5 011      derive clock from 352M PFD 100      derive clock from 396M PFD 101      derive clock from 540M PFD 110–111   Restricted
14–12 epdc_axi_podf	Divider for epdc_axi (SiPix) clock divider. Divider should be updated when output clock is gated. 000    divide by 1 001    divide by 2 010    divide by 3 (default) 011    divide by 4 100    divide by 5 101    divide by 6 110    divide by 7 111    divide by 8
11–9 -	This field is reserved. Reserved. Always set to 0.
8–6 pxp_axi_clk_sel	Selector for pxp_axi root clock multiplexer 000      derive clock from mmdc_ch0 clock 001      derive clock from pll3 010      derive clock from pll5 011      derive clock from 352M PFD

*Table continues on the next page...*

## CCM\_CHSCCDR field descriptions (continued)

Field	Description
	100      derive clock from 396M PFD 101      derive clock from 540M PFD 110-111   Restricted
5–3 pxp_axi_podf	Divider for pxp_axi clock divider. <b>NOTE:</b> Divider should be updated when output clock is gated.  000    divide by 1 001    divide by 2 010    divide by 3 (default) 011    divide by 4 100    divide by 5 101    divide by 6 110    divide by 7 111    divide by 8
2–0 -	This field is reserved. Reserved. Always set to 0.

## 15.6.15 CCM Serial Clock Divider Register 2 (CCM\_CSCDR2)

The figure below represents the CCM Serial Clock Divider Register 2(CSCDR2). The CSCDR2 register contains bits to control the clock generation sub-module dividers. The table below provides its field descriptions.

Address: 20C\_4000h base + 38h offset = 20C\_4038h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	Reserved							ecspi_clk_podf					ecspi_clk_sel	epdc_pix_clk_sel	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	epdc_pix_clk_sel	epdc_pix_pred			Reserved			lcdif_pix_clk_sel			lcdif_pix_pred			Reserved		
W																
Reset	1	0	0	1	1	0	1	1	0	1	0	0	1	0	0	0



## CCM\_CSCDR2 field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–25 -	This field is reserved. Reserved
24–19 ecspi_clk_podf	Divider for ecspi clock podf. <b>NOTE:</b> Divider should be updated when output clock is gated. <b>NOTE:</b> The input clock to this divider should be lower than 300Mhz, the predivider can be used to achieve this.  000000 divide by 1 111111 divide by 2 <sup>6</sup>
18 ecspi_clk_sel	Selector for the ECSPi clock multiplexor  0 pll3_60m 1 OSC clk
17–15 epdc_pix_clk_sel	Selector for epdc_pix root clock pre-multiplexer  000 derive clock from mmdc_ch0 clock 001 derive clock from pll3 010 derive clock from pll5 011 derive clock from 352M PFD 100 derive clock from 396M PFD 101 derive clock from 540M PFD (default) 110-111 Restricted
14–12 epdc_pix_pred	Divider for epdc_pix clock divider. <b>NOTE:</b> Divider should be updated when output clock is gated.  000 divide by 1 001 divide by 2 (default) 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
11–9 -	This field is reserved. Reserved. Always set to 0.
8–6 lcdif_pix_clk_sel	Selector for lcdif_pix root clock multiplexer  000 derive clock from mmdc_ch0 clock (default) 001 derive clock from pll3 010 derive clock from pll5 011 derive clock from 352M PFD 100 derive clock from 396M PFD 101 derive clock from 540M PFD 110-111 Restricted

Table continues on the next page...

**CCM\_CSCDR2 field descriptions (continued)**

Field	Description
5–3 lcdif_pix_pred	Divider for lcdif_pix clock divider. <b>NOTE:</b> Divider should be updated when output clock is gated.  000 divide by 1 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8 (default)
2–0 -	This field is reserved. Reserved. Always set to 0.

**15.6.16 CCM Serial Clock Divider Register 3 (CCM\_CSCDR3)**

The figure below represents the CCM Serial Clock Divider Register 3(CSCDR3). The CSCDR3 register contains bits to control the clock generation sub-module dividers. The table below provides its field descriptions.

Address: 20C\_4000h base + 3Ch offset = 20C\_403Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													lcdif_axi_podf		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	lcdif_axi_clk_sel		csi_core_podf			csi_core_clk_sel		Reserved								
W																
Reset	0	1	0	0	1	0	0	0	0	1	0	0	0	0	0	1

**CCM\_CSCDR3 field descriptions**

Field	Description
31–19 Reserved	This read-only field is reserved and always has the value 0.
18–16 lcdif_axi_podf	Divider for lcdif_axi clock. <b>NOTE:</b> Divider should be updated when output clock is gated.  000 divide by 1 001 divide by 2 (default) 010 divide by 3

Table continues on the next page...

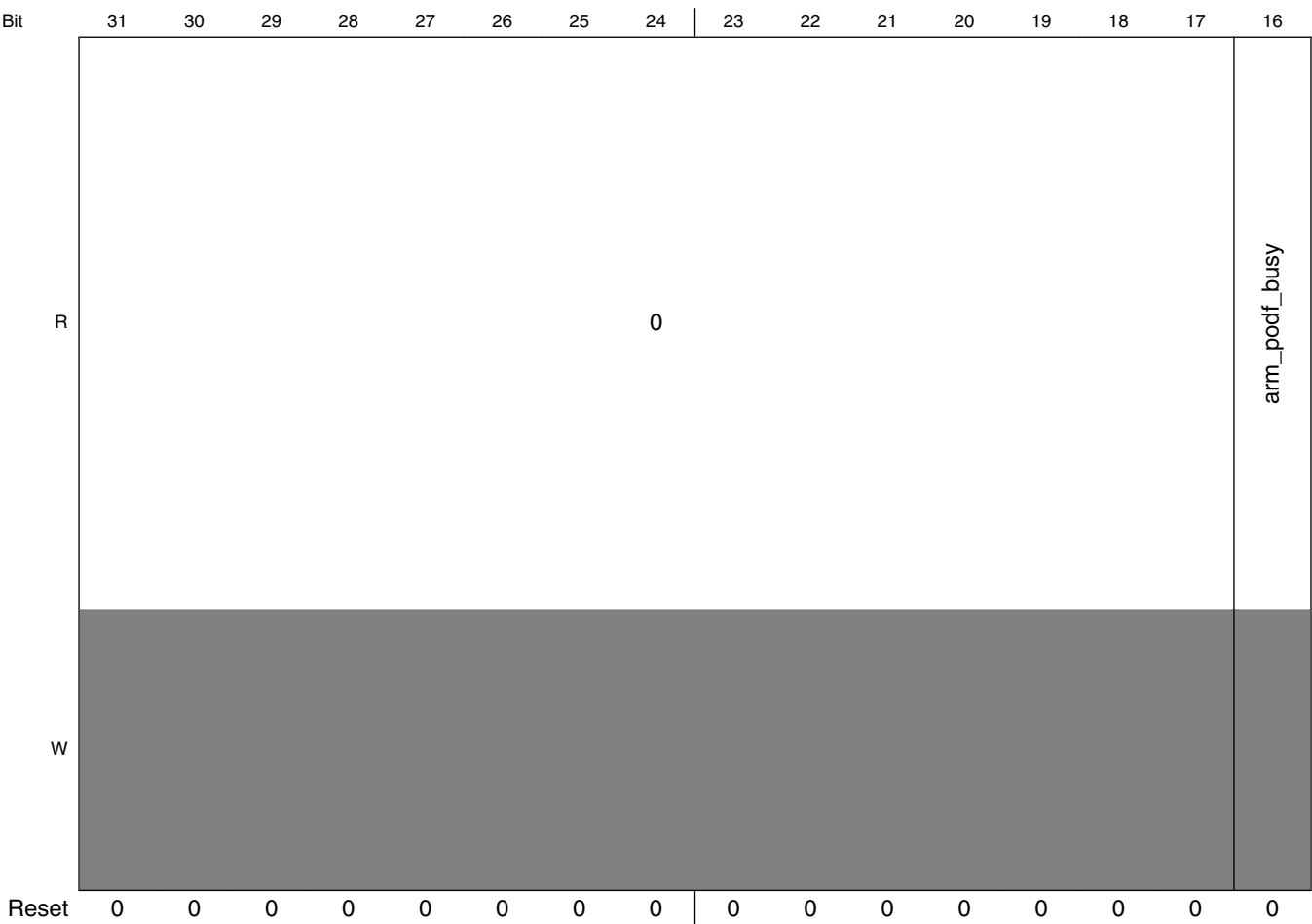
**CCM\_CSCDR3 field descriptions (continued)**

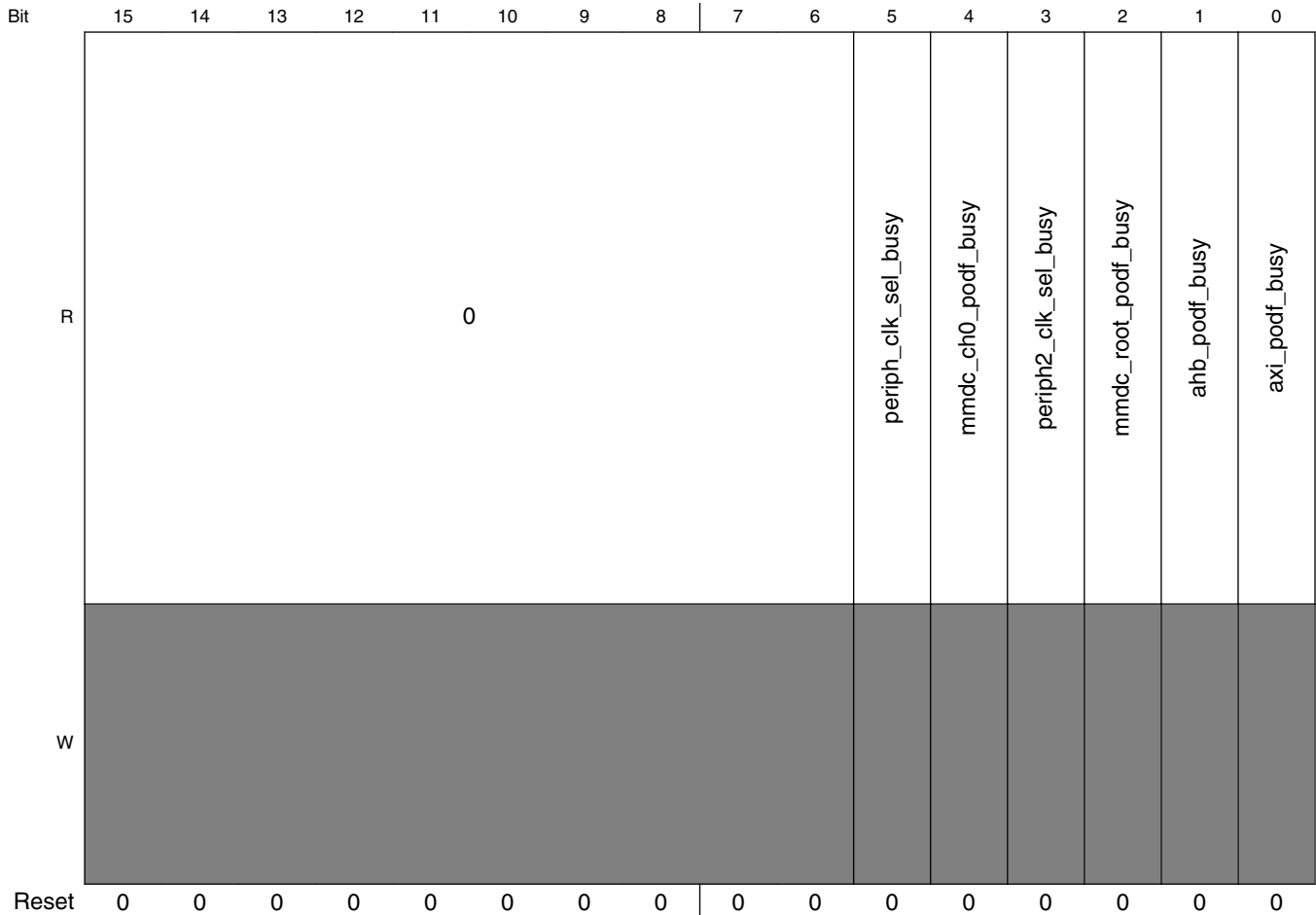
Field	Description
	011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
15–14 lcdif_axi_clk_sel	Selector for lcdif_axi clock multiplexer 00 derive clock from mmdc_ch0 clock (default) 01 derive clock from 396M PFD 10 derive clock from 120M 11 derive clock from 540M PFD
13–11 csi_core_podf	Post divider for csi_core clock. <b>NOTE:</b> Divider should be updated when output clock is gated. 000 divide by 1 001 divide by 2 (default) 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
10–9 csi_core_clk_sel	Selector for csi_core clock multiplexer 00 derive clock from mmdc_ch0 clock (default) 01 derive clock from 396M PFD 10 derive clock from 120M 11 derive clock from 540M PFD
8–0 -	This field is reserved. Reserved

### 15.6.17 CCM Divider Handshake In-Process Register (CCM\_CDHIPR)

The figure below represents the CCM Divider Handshake In-Process Register (CDHIPR). The CDHIPR register contains read-only bits that indicate that CCM is in the process of updating dividers or muxes that might need handshake with modules.

Address: 20C\_4000h base + 48h offset = 20C\_4048h





CCM\_CDHIPR field descriptions

Field	Description
31–17 Reserved	This read-only field is reserved and always has the value 0.
16 arm_podf_busy	Busy indicator for arm_podf. 0 divider is not busy and its value represents the actual division. 1 divider is busy with handshake process with module. The value read in the divider represents the previous value of division factor, and after the handshake the written value of the arm_podf will be applied.
15–6 Reserved	This read-only field is reserved and always has the value 0.
5 periph_clk_sel_busy	Busy indicator for periph_clk_sel mux control. 0 mux is not busy and its value represents the actual division. 1 mux is busy with handshake process with module. The value read in the periph_clk_sel represents the previous value of select, and after the handshake periph_clk_sel value will be applied.
4 mmdc_ch0_axi_podf_busy	Busy indicator for mmdc_ch0_axi_podf.

Table continues on the next page...

**CCM\_CDHIPR field descriptions (continued)**

Field	Description
	0 divider is not busy and its value represents the actual division. 1 divider is busy with handshake process with module. The value read in the divider represents the previous value odivision factor, and after the handshake the written value of the mmdc_ch0_axi_podf will be applied.
3 periph2_clk_sel_busy	Busy indicator for periph2_clk_sel mux control. 0 mux is not busy and its value represents the actual division. 1 mux is busy with handshake process with module. The value read in the periph2_clk_sel represents the previous value of select, and after the handshake periph2_clk_sel value will be applied.
2 mmdc_root_podf_busy	Busy indicator for mmdc_root_axi_podf. 0 divider is not busy and its value represents the actual division. 1 divider is busy with handshake process with module. The value read in the divider represents the previous value odivision factor, and after the handshake the written value of the mmdc_root_axi_podf will be applied.
1 ahb_podf_busy	Busy indicator for ahb_podf. 0 divider is not busy and its value represents the actual division. 1 divider is busy with handshake process with module. The value read in the divider represents the previous value odivision factor, and after the handshake the written value of the ahb_podf will be applied.
0 axi_podf_busy	Busy indicator for axi_podf. 0 divider is not busy and its value represents the actual division. 1 divider is busy with handshake process with module. The value read in the divider represents the previous value odivision factor, and after the handshake the written value of the axi_podf will be applied.

## 15.6.18 CCM Low Power Control Register (CCM\_CLPCR)

The figure below represents the CCM Low Power Control Register (CLPCR). The CLPCR register contains bits to control the low power modes operation. The table below provides its field descriptions.

Address: 20C\_4000h base + 54h offset = 20C\_4054h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				mask_l2cc_idle	mask_scu_idle	0			mask_core0_wfi	bypass_mmdc_root_lpm_hs	0	bypass_mmdc_ch0_lpm_hs	0	Reserved	wb_per_at_lpm
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				cosc_pwrdown	stby_count		VSTBY	dis_ref_osc	SBYOS	ARM_clk_dis_on_lpm	Reserved		bypass_pmic_ready	LPM	
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	1

**CCM\_CLPCR field descriptions**

Field	Description
31–28 Reserved	This read-only field is reserved and always has the value 0.
27 mask_l2cc_idle	Mask L2CC IDLE for entering low power mode. <b>NOTE:</b> Assertion of all bits[27:22] will generate low power mode request 1 L2CC IDLE is masked 0 L2CC IDLE is not masked
26 mask_scu_idle	Mask SCU IDLE for entering low power mode <b>NOTE:</b> Assertion of all bits[27:22] will generate low power mode request 1 SCU IDLE is masked 0 SCU IDLE is not masked

Table continues on the next page...

## CCM\_CLPCR field descriptions (continued)

Field	Description
25–23 Reserved	This read-only field is reserved and always has the value 0.
22 mask_core0_wfi	Mask WFI of core0 for entering low power mode <b>NOTE:</b> Assertion of all bits[27:22] will generate low power mode request 0 WFI of core0 is not masked 1 WFI of core0 is masked
21 bypass_mmdc_root_lpm_hs	Bypass handshake with mmdc_root on next entrance to low power mode (STOP or WAIT). CCM doesn't wait for the module's acknowledge. Handshake also will be bypassed, if CGR3 CG10 is set to gate fast mmdc_root clock. 0 handshake with mmdc_root on next entrance to low power mode will be performed. (default). 1 handshake with mmdc_root on next entrance to low power mode will be bypassed.
20 Reserved	This read-only field is reserved and always has the value 0.
19 bypass_mmdc_ch0_lpm_hs	Bypass handshake with mmdc_ch0 on next entrance to low power mode (STOP or WAIT). CCM doesn't wait for the module's acknowledge. Handshake will also be bypassed, if CGR3 CG10 is set to gate fast mmdc_ch0 clock. 0 Handshake with mmdc_ch0 on next entrance to low power mode will be performed. (default). 1 Handshake with mmdc_ch0 on next entrance to low power mode will be bypassed.
18 Reserved	This read-only field is reserved and always has the value 0.
17 -	This field is reserved. Reserved
16 wb_per_at_lpm	Enable periphery charge pump for well biasing at low power mode (stop or wait) 0 Periphery charge pump won't be enabled at STOP or WAIT low power modes 1 Periphery charge pump will be enabled at STOP or WAIT low power modes
15–12 Reserved	This read-only field is reserved and always has the value 0.
11 cosc_pwrdown	In run mode, software can manually control powering down of on chip oscillator, i.e. generating '1' on cosc_pwrdown signal. If software manually powered down the on chip oscillator, then sbyos functionality for on chip oscillator will be bypassed.  The manual closing of onchip oscillator should be performed only in case the reference oscillator is not the source of all the clocks generation. 0 On chip oscillator will not be powered down, i.e. cosc_pwrdown = '0'.(default) 1 On chip oscillator will be powered down, i.e. cosc_pwrdown = '1'.
10–9 stby_count	Standby counter definition. These two bits define, in the case of stop exit (if vstby bit was set), the amount of time CCM will wait between PMIC_VSTBY_REQ negation and the check of assertion of CCM_PMIC_READY . <b>NOTE:</b> Clock cycles ratio depends on pmic_delay_scaler, defined by CGPR[0] bit. 00 CCM will wait (1*pmic_delay_scaler)+1 ckil clock cycles 01 CCM will wait (3*pmic_delay_scaler)+1 ckil clock cycles 10 CCM will wait (7*pmic_delay_scaler)+1 ckil clock cycles 11 CCM will wait (15*pmic_delay_scaler)+1 ckil clock cycles

Table continues on the next page...



## CCM\_CLPCR field descriptions (continued)

Field	Description
8 VSTBY	<p>Voltage standby request bit. This bit defines if PMIC_VSTBY_REQ pin, which notifies external power management IC to move from functional voltage to standby voltage, will be asserted in STOP mode.</p> <p>0 Voltage will not be changed to standby voltage after next entrance to STOP mode. (PMIC_VSTBY_REQ will remain negated - '0')</p> <p>1 Voltage will be requested to change to standby voltage after next entrance to stop mode. (PMIC_VSTBY_REQ will be asserted - '1').</p>
7 dis_ref_osc	<p>dis_ref_osc - in run mode, software can manually control closing of external reference oscillator clock, i.e. generating '1' on CCM_REF_EN_B signal. If software closed manually the external reference clock, then sbyos functionality will be bypassed.</p> <p>The manual closing of external reference oscillator should be performed only in case the reference oscillator is not the source of any clock generation.</p> <p><b>NOTE:</b> When returning from stop mode, the PMIC_VSTBY_REQ will be deasserted (if it was asserted when entering stop mode), and CCM will wait for indication that functional voltage is ready (by sampling the assertion of pmic_vfunctional_ready) before continuing the process of exiting from stop mode. See stby_count bits.</p> <p>0 external high frequency oscillator will be enabled, i.e. CCM_REF_EN_B = '0'.(default)</p> <p>1 external high frequency oscillator will be disabled, i.e. CCM_REF_EN_B = '1'</p>
6 SBYOS	<p>Standby clock oscillator bit. This bit defines if cosc_pwrdown, which power down the on chip oscillator, will be asserted in STOP mode. This bit is discarded if cosc_pwrdown='1' for the on chip oscillator.</p> <p>0 On-chip oscillator will not be powered down, after next entrance to STOP mode. (CCM_REF_EN_B will remain asserted - '0' and cosc_pwrdown will remain de asserted - '0')</p> <p>1 On-chip oscillator will be powered down, after next entrance to STOP mode. (CCM_REF_EN_B will be deasserted - '1' and cosc_pwrdown will be asserted - '1'). (Default.) When returning from STOP mode, external oscillator will be enabled again, on-chip oscillator will return to oscillator mode, and after oscnt count, CCM will continue with the exit from the STOP mode process.</p>
5 ARM_clk_dis_on_lpm	<p>Define if ARM clocks (arm_clk, soc_mxclk, soc_pclk, soc_dbg_pclk, vl_wrck) will be disabled on wait mode. This is useful for debug mode, when the user still wants to simulate entering wait mode and still keep ARM clock functioning.</p> <p><b>NOTE:</b> Software should not enable ARM power gating in wait mode if this bit is cleared.</p> <p>0 ARM clock enabled on wait mode.</p> <p>1 ARM clock disabled on wait mode. (default).</p>
4-3 -	<p>This field is reserved. Reserved</p>
2 bypass_pmic_ready	<p>By asserting this bit CCM will bypass waiting for CCM_PMIC_READY signal when coming out of STOP mode. This should be used for PMIC's that don't support the CCM_PMIC_READY signal.</p> <p>0 Don't bypass the CCM_PMIC_READY signal - CCM will wait for it's assertion during exit of low power mode if standby voltage was enabled.</p> <p>1 bypass the CCM_PMIC_READY signal - CCM will not wait for it's assertion during exit of low power mode if standby voltage was enabled.</p>
1-0 LPM	<p>Setting the low power mode that system will enter on next assertion of dsm_request signal.</p> <p>00 Remain in run mode</p> <p>01 Transfer to wait mode</p> <p>10 Transfer to stop mode</p> <p>11 Reserved</p>

15.6.19 CCM Interrupt Status Register (CCM\_CISR)

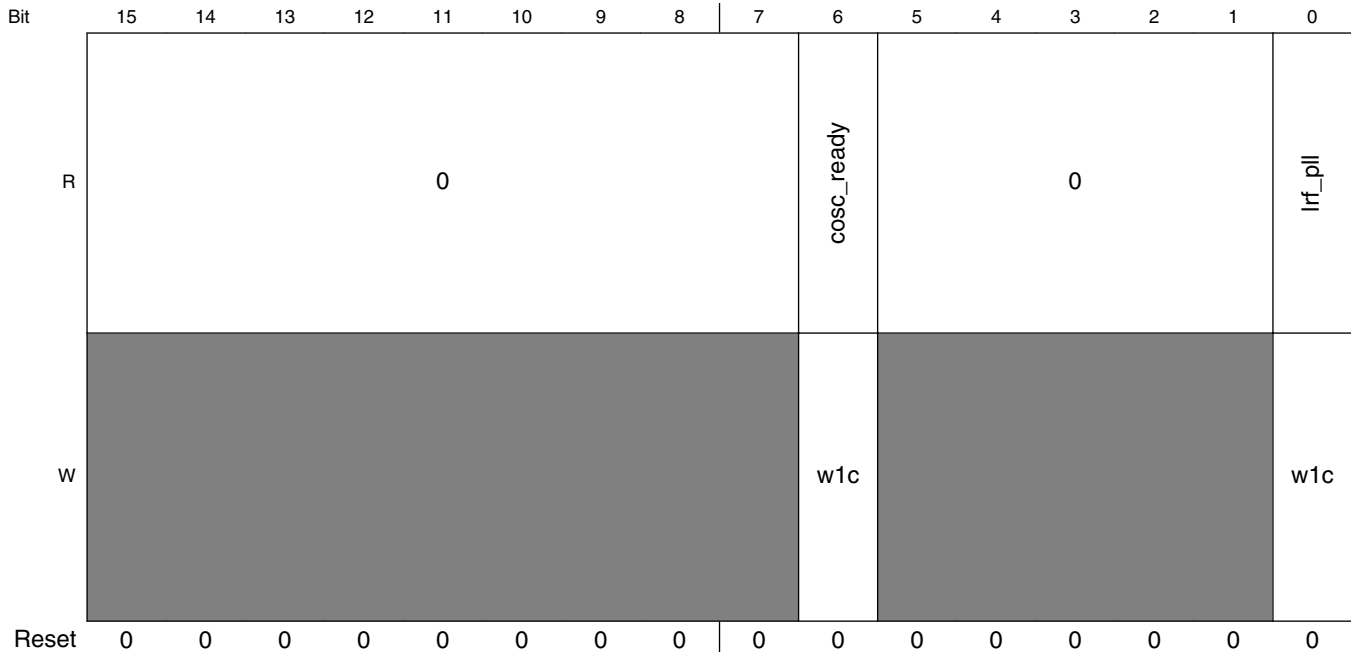
The figure below represents the CCM Interrupt Status Register (CISR). This is a write one to clear register. Once a interrupt is generated, software should write one to clear it. The table below provides its field descriptions.

NOTE

CCM interrupt request 1 can be masked by CCM interrupt request 1 mask bit. CCM interrupt request 2 can be masked by CCM interrupt request 2 mask bit.

Address: 20C\_4000h base + 58h offset = 20C\_4058h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					arm_podf_loaded	0		mmdc_ch0_podf_loaded	periph_clk_sel_loaded	mmdc_root_podf_loaded	ahb_podf_loaded	periph2_clk_sel_loaded	0	axi_podf_loaded	0
W						w1c			w1c	w1c	w1c	w1c	w1c		w1c	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



CCM\_CISR field descriptions

Field	Description
31–27 Reserved	This read-only field is reserved and always has the value 0.
26 arm_podf_loaded	CCM interrupt request 1 generated due to frequency change of arm_podf. The interrupt will commence only if arm_podf is loaded during a arm dvfs operation.  0 interrupt is not generated due to frequency change of arm_podf 1 interrupt generated due to frequency change of arm_podf
25–24 Reserved	This read-only field is reserved and always has the value 0.
23 mmdc_ch0_podf_loaded	CCM interrupt request 1 generated due to update of mmdc_ch0_axi_podf.  0 interrupt is not generated due to update of mmdc_ch0_axi_podf. 1 interrupt generated due to update of mmdc_ch0_axi_podf*
22 periph_clk_sel_loaded	CCM interrupt request 1 generated due to update of periph_clk_sel.  0 interrupt is not generated due to update of periph_clk_sel. 1 interrupt generated due to update of periph_clk_sel.
21 mmdc_root_podf_loaded	CCM interrupt request 1 generated due to frequency change of mmdc_ch0_podf_ <b>loaded</b>  0 interrupt is not generated due to frequency change of mmdc_ch0_podf_ <b>loaded</b> 1 interrupt generated due to frequency change of mmdc_ch0_podf_ <b>loaded</b>
20 ahb_podf_loaded	CCM interrupt request 1 generated due to frequency change of ahb_podf  0 interrupt is not generated due to frequency change of ahb_podf 1 interrupt generated due to frequency change of ahb_podf
19 periph2_clk_sel_loaded	CCM interrupt request 1 generated due to frequency change of periph2_clk_sel

Table continues on the next page...

**CCM\_CISR field descriptions (continued)**

Field	Description
	0 interrupt is not generated due to frequency change of periph2_clk_sel 1 interrupt generated due to frequency change of periph2_clk_sel
18 Reserved	This read-only field is reserved and always has the value 0.  0 interrupt is not generated due to frequency change of mmdc_ch0_axi_podf 1 interrupt generated due to frequency change of mmdc_ch0_axi_podf
17 axi_podf_loaded	CCM interrupt request 1 generated due to frequency change of axi_podf  0 interrupt is not generated due to frequency change of axi_podf 1 interrupt generated due to frequency change of axi_podf
16–7 Reserved	This read-only field is reserved and always has the value 0.
6 cosc_ready	CCM interrupt request 2 generated due to on board oscillator ready, i.e. oscnt has finished counting.  0 interrupt is not generated due to on board oscillator ready 1 interrupt generated due to on board oscillator ready
5–1 Reserved	This read-only field is reserved and always has the value 0.
0 lrf_pll	CCM interrupt request 2 generated due to lock of all enabled and not bypassed PLLs  0 interrupt is not generated due to lock ready of all enabled and not bypassed PLLs 1 interrupt generated due to lock ready of all enabled and not bypassed PLLs

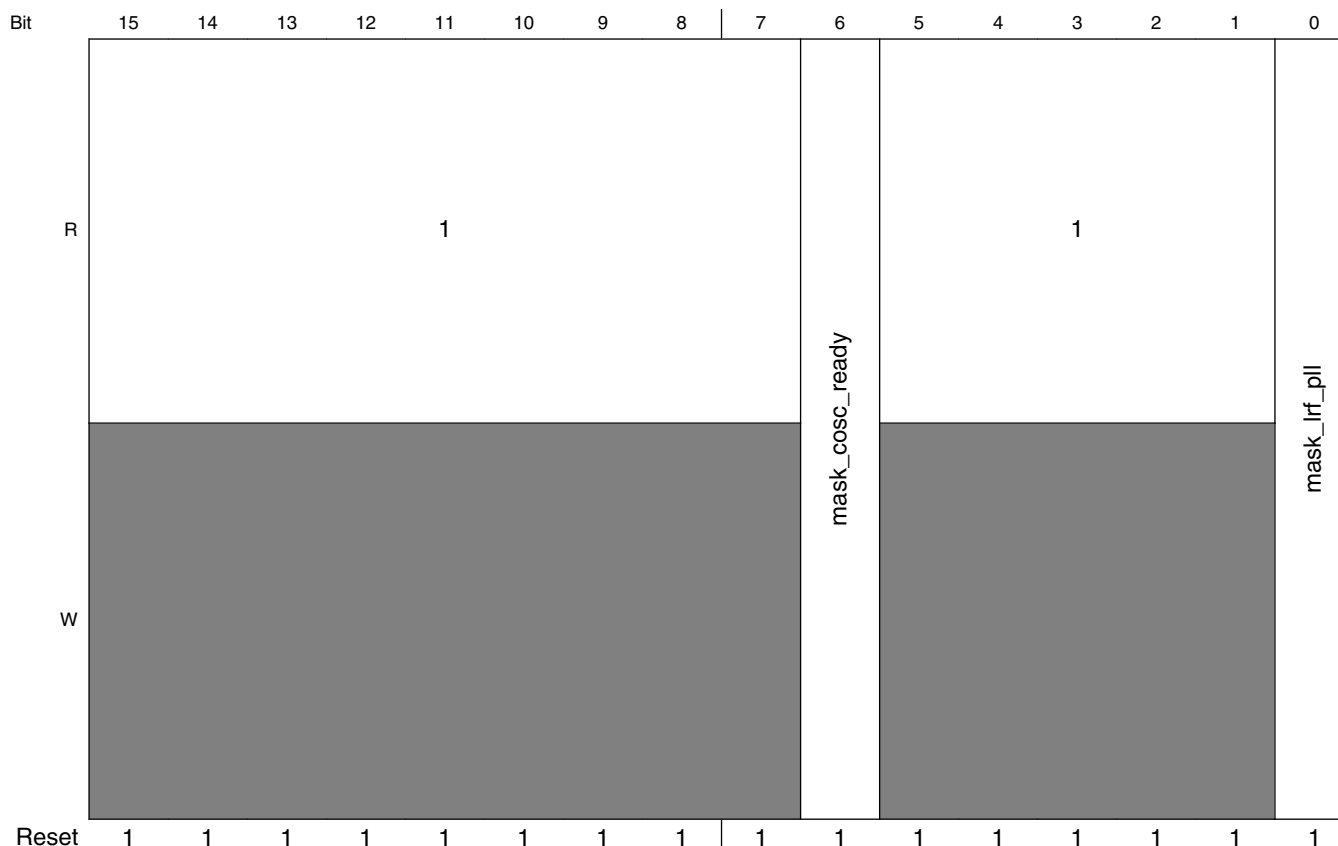
15.6.20 CCM Interrupt Mask Register (CCM\_CIMR)

The figure below represents the CCM Interrupt Mask Register (CIMR). The table below provides its field descriptions.

Address: 20C\_4000h base + 5Ch offset = 20C\_405Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16										
R	1					arm_podf_loaded			mask_mmdc_ch0_podf_loaded			mask_periph_clk_sel_loaded		mask_mmdc_root_podf_loaded		mask_ahb_podf_loaded		mask_periph2_clk_sel_loaded		mask_mmdc_ch0_axi_podf_loaded		mask_axi_podf_loaded		1		
W																										
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

## CCM Memory Map/Register Definition



### CCM\_CIMR field descriptions

Field	Description
31–27 Reserved	This read-only field is reserved and always has the value 1.
26 arm_podf_loaded	mask interrupt generation due to frequency change of arm_podf 0 don't mask interrupt due to frequency change of arm_podf - interrupt will be created 1 mask interrupt due to frequency change of arm_podf
25–24 Reserved	This read-only field is reserved and always has the value 1.
23 mask_mmdc_ch0_podf_loaded	mask interrupt generation due to update of mask_mmdc_ch0_podf 0 don't mask interrupt due to update of mask_mmdc_ch0_podf - interrupt will be created 1 mask interrupt due to update of mask_mmdc_ch0_podf
22 mask_periph_clk_sel_loaded	mask interrupt generation due to update of periph_clk_sel. 0 don't mask interrupt due to update of periph_clk_sel - interrupt will be created 1 mask interrupt due to update of periph_clk_sel
21 mask_mmdc_root_podf_loaded	mask interrupt generation due to update of mask_mmdc_root_podf 0 don't mask interrupt due to update of mask_mmdc_root_podf - interrupt will be created 1 mask interrupt due to update of mask_mmdc_root_podf

Table continues on the next page...

**CCM\_CIMR field descriptions (continued)**

<b>Field</b>	<b>Description</b>
20 mask_ahb_podf_loaded	mask interrupt generation due to frequency change of ahb_podf 0 don't mask interrupt due to frequency change of ahb_podf - interrupt will be created 1 mask interrupt due to frequency change of ahb_podf
19 mask_periph2_clk_sel_loaded	mask interrupt generation due to update of periph2_clk_sel. 0 don't mask interrupt due to update of periph2_clk_sel - interrupt will be created 1 mask interrupt due to update of periph2_clk_sel
18 mask_mmdc_ch0_axi_podf_loaded	mask interrupt generation due to frequency change of mmdc_ch0_axi_podf 0 don't mask interrupt due to frequency change of mmdc_ch0_axi_podf - interrupt will be created 1 mask interrupt due to frequency change of mmdc_ch0_axi_podf
17 mask_axi_podf_loaded	mask interrupt generation due to frequency change of axi_podf 0 don't mask interrupt due to frequency change of axi_podf - interrupt will be created 1 mask interrupt due to frequency change of axi_podf
16–7 Reserved	This read-only field is reserved and always has the value 1.
6 mask_cosc_ready	mask interrupt generation due to on board oscillator ready 0 don't mask interrupt due to on board oscillator ready - interrupt will be created 1 mask interrupt due to on board oscillator ready
5–1 Reserved	This read-only field is reserved and always has the value 1.
0 mask_lrf_pll	mask interrupt generation due to lrf of PLLs 0 don't mask interrupt due to lrf of PLLs - interrupt will be created 1 mask interrupt due to lrf of PLLs

## 15.6.21 CCM Clock Output Source Register (CCM\_CCOSR)

The figure below represents the CCM Clock Output Source Register (CCOSR). The CCOSR register contains bits to control the clock that will be generated on the output ipp\_do\_clk01 (CCM\_CLKO). The table below provides its field descriptions.

Address: 20C\_4000h base + 60h offset = 20C\_4060h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							CLKO2_EN	CLKO2_DIV				CLKO2_SEL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							CLK_OUT_SEL	CLKO1_EN	CLKO1_DIV				CLKO_SEL		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

**CCM\_CCOSR field descriptions**

Field	Description
31–25 Reserved	This read-only field is reserved and always has the value 0.
24 CLKO2_EN	Enable of CCM_CLKO2 clock 0 CCM_CLKO2 disabled. 1 CCM_CLKO2 enabled.
23–21 CLKO2_DIV	Setting the divider of CCM_CLKO2 000 divide by 1 (default) 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
20–16 CLKO2_SEL	Selection of the clock to be generated on CCM_CLKO2 00000 mmdc_ch0_axi_clk_root 00001 mmdc_root_axi_clk_root

*Table continues on the next page...*



**CCM\_CCOSR field descriptions (continued)**

Field	Description
	00010 usdhc4_clk_root 00011 usdhc1_clk_root 00101 wrck_clk_root 00110 ecspi_clk_root 01000 usdhc3_clk_root 01001 pcie_clk_root 01010 arm_clk_root (default) 01011 csi_core 01100 lcdif_axi 01110 osc_clk 10000 gpu2d_ovg_core_clk_root 10001 usdhc2_clk_root 10010 ssi1_clk_root 10011 ssi2_clk_root 10100 ssi3_clk_root 10101 gpu2d_core_clk_root 11010 extern_audio_clk_root 11011 aclk_eim_slow_clk_root 11100 uart_clk_root 11101 spdif0_clk_root 11110 spdif1_clk_root 11111 Reserved
15–9 Reserved	This read-only field is reserved and always has the value 0.
8 CLK_OUT_SEL	CCM_CLKO1 output to reflect CCM_CLKO1 or CCM_CLKO2 clocks 0 CCM_CLKO1 output drives CCM_CLKO1 clock 1 CCM_CLKO1 output drives CCM_CLKO2 clock
7 CLKO1_EN	Enable of CCM_CLKO1 clock 0 CCM_CLKO1 disabled. 1 CCM_CLKO1 enabled.
6–4 CLKO1_DIV	Setting the divider of CCM_CLKO1 000 divide by 1(default) 001 divide by 2 010 divide by 3 011 divide by 4 100 divide by 5 101 divide by 6 110 divide by 7 111 divide by 8
3–0 CLKO_SEL	Selection of the clock to be generated on CCM_CLKO1 0000 pll3_sw_clk (this inputs has additional constant division /2) 0001 pll2_main_clk (default) (this inputs has additional constant division /2) 0010 pll1_main_clk (this inputs has additional constant division /2)

*Table continues on the next page...*

## CCM\_CCOSR field descriptions (continued)

Field	Description
0011	pll5_main_clk (this inputs has additional constant division /2)
0101	ocram_clk_root
0111	pxp_axi_clk_root
1000	epdc_axi_clk_root
1001	lcdif_pix_clk_root
1010	epdc_pix_clk_root
1011	ahb_clk_root
1100	ipg_clk_root
1101	perclk_root
1110	ckil_sync_clk_root
1111	pll4_main_clk

## 15.6.22 CCM General Purpose Register (CCM.CGPR)

Fast PLL enable. Can be used to engage PLL faster after STOP mode, if 24MHz OSC was active

Address: 20C\_4000h base + 64h offset = 20C\_4064h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															FPL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	1							0		1		efuse_prog_supply_gate	Reserved	mmdc_ext_clk_dis	pmic_delay_scaler	
W															1	
Reset	1	1	1	1	1	1	1	0	0	1	1	0	0	0	1	0

**CCM\_CGPR field descriptions**

Field	Description
31–17 Reserved	This read-only field is reserved and always has the value 0.
16 FPL	Fast PLL enable.  0 Engage PLL enable default way. 1 Engage PLL enable 3 CKIL clocks earlier at exiting low power mode (STOP). Should be used only if 24MHz OSC was active in low power mode.
15–9 Reserved	This read-only field is reserved and always has the value 1.
8–7 Reserved	This read-only field is reserved and always has the value 0.
6–5 Reserved	This read-only field is reserved and always has the value 1.
4 efuse_prog_ supply_gate	Defines the value of the output signal cgpr_dout[4]. Gate of program supply for efuse programing  0 fuse programing supply voltage is gated off to the efuse module 1 allow fuse programing.
3 -	This field is reserved. Reserved
2 mmdc_ext_clk_ dis	Disable external clock driver of MMDC during STOP mode  1 disable during stop mode 0 don't disable during stop mode.
1 -	Reserved. Keep default value set to '1' for proper operation.
0 pmic_delay_ scaler	Defines clock division of clock for stby_count (pmic delay counter)  0 clock is not divided 1 clock is divided /8

**15.6.23 CCM Clock Gating Register 0 (CCM\_CCGR0)**

CG(i) bits CCGR 0-6

These bits are used to turn on/off the clock to each module independently. The following table details the possible clock activity conditions for each module.

CGR value	Clock Activity Description
00	Clock is off during all modes. Stop enter hardware handshake is disabled.
01	Clock is on in run mode, but off in WAIT and STOP modes
10	Not applicable (Reserved).
11	Clock is on during all modes, except STOP mode.

Module should be stopped, before set its bits to "0"; clocks to the module will be stopped immediately.

The tables above show the register mappings for the different CGRs. The clock connectivity table should be used to match the "CCM output affected" to the actual clocks going into the modules.

The figure below represents the CCM Clock Gating Register 0 (CCM\_CCGR0). The clock gating Registers define the clock gating for power reduction of each clock (CG(i) bits). There are 7 CGR registers. The number of registers required is according to the number of peripherals in the system.

Address: 20C\_4000h base + 68h offset = 20C\_4068h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

### CCM\_CCGR0 field descriptions

Field	Description
31–30 CG15	Reserved
29–28 CG14	Reserved
27–26 CG13	Reserved
25–24 CG12	Reserved
23–22 CG11	CPU debug clocks (arm_dbg_clk_enable)
21–20 CG10	Reserved
19–18 CG9	Reserved
17–16 CG8	Reserved
15–14 CG7	Reserved
13–12 CG6	Reserved
11–10 CG5	Reserved
9–8 CG4	Reserved

Table continues on the next page...

**CCM\_CCGR0 field descriptions (continued)**

Field	Description
7–6 CG3	Reserved
5–4 CG2	Reserved
3–2 CG1	aips_tz2 clocks (aips_tz2_clk_enable)
1–0 CG0	aips_tz1 clocks (aips_tz1_clk_enable)

**15.6.24 CCM Clock Gating Register 1 (CCM\_CCGR1)**

The figure below represents the CCM Clock Gating Register 1 (CCM\_CCGR1). The clock gating registers define the clock gating for power reduction of each clock (CG(i) bits). There are 8 CGR registers. The number of registers required is determined by the number of peripherals in the system.

Address: 20C\_4000h base + 6Ch offset = 20C\_406Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
	CG15	CG14	CG13	CG12	CG11	CG10	CG9	CG8								
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
	CG7	CG6	CG5	CG4	CG3	CG2	CG1	CG0								
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

**CCM\_CCGR1 field descriptions**

Field	Description
31–30 CG15	Reserved
29–28 CG14	Reserved
27–26 CG13	gpu2d ovg clock (gpu2d_ovg_core_clk_enable)
25–24 CG12	Reserved
23–22 CG11	gpt serial clock (gpt_serial_clk_enable)
21–20 CG10	gpt bus clock (gpt_clk_enable)

*Table continues on the next page...*

**CCM\_CCGR1 field descriptions (continued)**

Field	Description
19–18 CG9	Reserved
17–16 CG8	esai clocks (extern_audio_clk_enable)
15–14 CG7	epit2 clocks (epit2_clk_enable)
13–12 CG6	epit1 clocks (epit1_clk_enable)
11–10 CG5	Reserved
9–8 CG4	Reserved
7–6 CG3	ecspi4 clocks (ecspi4_clk_enable)
5–4 CG2	ecspi3 clocks (ecspi3_clk_enable)
3–2 CG1	ecspi2 clocks (ecspi2_clk_enable)
1–0 CG0	ecspi1 clocks (ecspi1_clk_enable)

**15.6.25 CCM Clock Gating Register 2 (CCM\_CCGR2)**

The figure below represents the CCM Clock Gating Register 2 (CCM\_CCGR2). The clock gating registers define the clock gating for power reduction of each clock (CG(i) bits). There are 8 CGR registers. The number of registers required is determined by the number of peripherals in the system.

Address: 20C\_4000h base + 70h offset = 20C\_4070h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
Reset	1	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

**CCM\_CCGR2 field descriptions**

<b>Field</b>	<b>Description</b>
31–30 CG15	Reserved
29–28 CG14	Reserved
27–26 CG13	ipsync_vdoa_ipg clocks (ipsync_vdoa_ipg_master_clk_enable)
25–24 CG12	Reserved
23–22 CG11	ipsync_ip2apb_tzasc1_ipg clocks (ipsync_ip2apb_tzasc1_ipg_master_clk_enable)
21–20 CG10	ipmux3 clock (ipmux3_clk_enable)
19–18 CG9	ipmux2 clock (ipmux2_clk_enable)
17–16 CG8	ipmux1 clock (ipmux1_clk_enable)
15–14 CG7	iomux_ipt_clk_io clock (iomux_ipt_clk_io_enable)
13–12 CG6	OCOTP_CTRL clock (iim_clk_enable)
11–10 CG5	i2c3_serial clock (i2c3_serial_clk_enable)
9–8 CG4	i2c2_serial clock (i2c2_serial_clk_enable)
7–6 CG3	i2c1_serial clock (i2c1_serial_clk_enable)
5–4 CG2	Reserved
3–2 CG1	Reserved
1–0 CG0	Reserved

## 15.6.26 CCM Clock Gating Register 3 (CCM\_CCGR3)

The figure below represents the CCM Clock Gating Register 3 (CCM\_CCGR3). The clock gating Registers define the clock gating for power reduction of each clock (CG(i) bits). There are 8 CGR registers. The number of registers required is determined by the number of peripherals in the system.

Address: 20C\_4000h base + 74h offset = 20C\_4074h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

### CCM\_CCGR3 field descriptions

Field	Description
31–30 CG15	Reserved
29–28 CG14	ocram clock (ocram_clk_enable)
27–26 CG13	mmdc_core_ipg_clk_p1 clock (mmdc_core_ipg_clk_p1_enable)
25–24 CG12	mmdc_core_ipg_clk_p0 clock (mmdc_core_ipg_clk_p0_enable)
23–22 CG11	Reserved
21–20 CG10	mmdc_core_aclk_fast_core_p0 clock (mmdc_core_aclk_fast_core_p0_enable)
19–18 CG9	Reserved
17–16 CG8	Reserved
15–14 CG7	Reserved
13–12 CG6	Reserved
11–10 CG5	epdc_pix clock (epdc_pix_clk_enable)
9–8 CG4	lcdif_pix clock (lcdif_pix_clk_enable)

Table continues on the next page...



**CCM\_CCGR3 field descriptions (continued)**

Field	Description
7–6 CG3	lcdif axi clock (lcdif_axi_clk_enable)
5–4 CG2	epdc axi clock (epdc_axi_clk_enable)
3–2 CG1	pxp axi clock (pxp_axi_clk_enable)
1–0 CG0	csi core clock (csi_core_clk_enable)

**15.6.27 CCM Clock Gating Register 4 (CCM\_CCGR4)**

The figure below represents the CCM Clock Gating Register 4 (CCM\_CCGR4). The clock gating Registers define the clock gating for power reduction of each clock (CG(i) bits). There are 8 CGR registers. The number of registers required is determined by the number of peripherals in the system.

Address: 20C\_4000h base + 78h offset = 20C\_4078h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

**CCM\_CCGR4 field descriptions**

Field	Description
31–30 CG15	rawnand_u_gpmi_input_apb clock (rawnand_u_gpmi_input_apb_clk_enable)
29–28 CG14	rawnand_u_gpmi_bch_input_gpmi_io clock (rawnand_u_gpmi_bch_input_gpmi_io_clk_enable)
27–26 CG13	rawnand_u_gpmi_bch_input_bch clock (rawnand_u_gpmi_bch_input_bch_clk_enable)
25–24 CG12	rawnand_u_bch_input_apb clock (rawnand_u_bch_input_apb_clk_enable)
23–22 CG11	pwm4 clocks (pwm4_clk_enable)
21–20 CG10	pwm3 clocks (pwm3_clk_enable)

*Table continues on the next page...*

**CCM\_CCGR4 field descriptions (continued)**

Field	Description
19–18 CG9	pwm2 clocks (pwm2_clk_enable)
17–16 CG8	pwm1 clocks (pwm1_clk_enable)
15–14 CG7	pl301_mx6qper2_mainclk_enable (pl301_mx6qper2_mainclk_enable)
13–12 CG6	Reserved
11–10 CG5	Reserved
9–8 CG4	pl301_mx6qfast1_s133 clock (pl301_mx6qfast1_s133clk_enable)
7–6 CG3	Reserved.
5–4 CG2	Reserved.
3–2 CG1	Reserved.
1–0 CG0	Reserved

**15.6.28 CCM Clock Gating Register 5 (CCM\_CCGR5)**

The figure below represents the CCM Clock Gating Register 5 (CCM\_CCGR5). The clock gating Registers define the clock gating for power reduction of each clock (CG(i) bits). There are 8 CGR registers. The number of registers required is determined by the number of peripherals in the system.

Address: 20C\_4000h base + 7Ch offset = 20C\_407Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

**CCM\_CCGR5 field descriptions**

<b>Field</b>	<b>Description</b>
31–30 CG15	Reserved
29–28 CG14	Reserved
27–26 CG13	uart_serial clock (uart_serial_clk_enable)
25–24 CG12	uart clock (uart_clk_enable)
23–22 CG11	ssi3 clocks (ssi3_clk_enable)
21–20 CG10	ssi2 clocks (ssi2_clk_enable)
19–18 CG9	ssi1 clocks (ssi1_clk_enable)
17–16 CG8	Reserved
15–14 CG7	spdif clock (spdif_clk_enable)
13–12 CG6	spba clock (spba_clk_enable)
11–10 CG5	Reserved
9–8 CG4	Reserved
7–6 CG3	sdma clock (sdma_clk_enable)
5–4 CG2	100M clock (100M_clk_enable)
3–2 CG1	Reserved
1–0 CG0	rom clock (rom_clk_enable)

## 15.6.29 CCM Clock Gating Register 6 (CCM\_CCGR6)

The figure below represents the CCM Clock Gating Register 6 (CCM\_CCGR6). The clock gating Registers define the clock gating for power reduction of each clock (CG(i) bits). There are 8 CGR registers. The number of registers required is determined by the number of peripherals in the system.

Address: 20C\_4000h base + 80h offset = 20C\_4080h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
	CG15		CG14		CG13		CG12		CG11		CG10		CG9		CG8	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
	CG7		CG6		CG5		CG4		CG3		CG2		CG1		CG0	
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

### CCM\_CCGR6 field descriptions

Field	Description
31–30 CG15	Reserved
29–28 CG14	Reserved
27–26 CG13	Reserved
25–24 CG12	Reserved
23–22 CG11	Reserved
21–20 CG10	Reserved
19–18 CG9	Reserved
17–16 CG8	Reserved
15–14 CG7	vpu clocks (vpu_clk_enable)
13–12 CG6	vdoaxicl root clock (vdoaxicl_clk_enable)
11–10 CG5	eim_slow clocks (eim_slow_clk_enable)
9–8 CG4	usdhc4 clocks (usdhc4_clk_enable)

Table continues on the next page...

**CCM\_CCGR6 field descriptions (continued)**

Field	Description
7–6 CG3	usdhc3 clocks (usdhc3_clk_enable)
5–4 CG2	usdhc2 clocks (usdhc2_clk_enable)
3–2 CG1	usdhc1 clocks (usdhc1_clk_enable)
1–0 CG0	usboh3 clock (usboh3_clk_enable)

**15.6.30 CCM Module Enable Override Register (CCM\_CMEOR)**

The follow figure represents the CCM Module Enable Override Register (CMEOR). The CMEOR register contains bits to override the clock enable signal from the module. This bit will be applicable only for modules whose clock enable signals are used. The following table provides its field descriptions.

Address: 20C\_4000h base + 88h offset = 20C\_4088h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	1															
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	1					mod_en_ov_gpu2d	1		mod_en_usdhc	mod_en_ov_epit	mod_en_ov_gpt	1				
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

**CCM\_CMEOR field descriptions**

Field	Description
31–11 Reserved	This read-only field is reserved and always has the value 1.
10 mod_en_ov_gpu2d	Override clock enable signal from GPU2D - clock will not be gated based on GPU2D's signal 'gpu2d_busy'.

*Table continues on the next page...*

**CCM\_CMEOR field descriptions (continued)**

Field	Description
	0 don't override module enable signal 1 override module enable signal
9–8 Reserved	This read-only field is reserved and always has the value 1.
7 mod_en_usdhc	override clock enable signal from USDHC. 0 don't override module enable signal 1 override module enable signal
6 mod_en_ov_epit	Override clock enable signal from EPIT - clock will not be gated based on EPIT's signal 'ipg_enable_clk' . 0 don't override module enable signal 1 override module enable signal
5 mod_en_ov_gpt	Override clock enable signal from GPT - clock will not be gated based on GPT's signal 'ipg_enable_clk' . 0 don't override module enable signal 1 override module enable signal
4–0 Reserved	This read-only field is reserved and always has the value 1.

## 15.7 CCM Analog Memory Map/Register Definition

This section describes the registers for the analog PLLs. The registers which have the same description are grouped within { }. The register offsets for the various PLLs are:

- ARM PLL: {0h000, 0h004, 0h008, 0h00C}.
- USB1 PLL: {0h010, 0h014, 0h018, 0h01C}, {0h0F0, 0h0F4, 0h0F8, 0h0FC}.
- System PLL: {0h030, 0h034, 0h038, 0h03C}, 0h040, 0h050, 0h060, {0h100, 0h104, 0h108, 0h10C}.
- Audio / Video PLL: {0h070, 0h074, 0h078, 0h07C}, 0h080, 0h090, {0h0A0, 0h0A4, 0h0A8, 0h0AC}, 0h0B0, 0h0C0

**CCM\_ANALOG memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20C_8000	Analog ARM PLL control Register (CCM_ANALOG_PLL_ARM)	32	R/W	0001_3042h	<a href="#">15.7.1/581</a>
20C_8004	Analog ARM PLL control Register (CCM_ANALOG_PLL_ARM_SET)	32	R/W	0001_3042h	<a href="#">15.7.1/581</a>
20C_8008	Analog ARM PLL control Register (CCM_ANALOG_PLL_ARM_CLR)	32	R/W	0001_3042h	<a href="#">15.7.1/581</a>

*Table continues on the next page...*

**CCM\_ANALOG memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/ page</b>
20C_800C	Analog ARM PLL control Register (CCM_ANALOG_PLL_ARM_TOG)	32	R/W	0001_3042h	<a href="#">15.7.1/581</a>
20C_8010	Analog USB1 480MHz PLL Control Register (CCM_ANALOG_PLL_USB1)	32	R/W	0001_2000h	<a href="#">15.7.2/583</a>
20C_8014	Analog USB1 480MHz PLL Control Register (CCM_ANALOG_PLL_USB1_SET)	32	R/W	0001_2000h	<a href="#">15.7.2/583</a>
20C_8018	Analog USB1 480MHz PLL Control Register (CCM_ANALOG_PLL_USB1_CLR)	32	R/W	0001_2000h	<a href="#">15.7.2/583</a>
20C_801C	Analog USB1 480MHz PLL Control Register (CCM_ANALOG_PLL_USB1_TOG)	32	R/W	0001_2000h	<a href="#">15.7.2/583</a>
20C_8020	Analog USB2 480MHz PLL Control Register (CCM_ANALOG_PLL_USB2)	32	R/W	0001_2000h	<a href="#">15.7.3/585</a>
20C_8024	Analog USB2 480MHz PLL Control Register (CCM_ANALOG_PLL_USB2_SET)	32	R/W	0001_2000h	<a href="#">15.7.3/585</a>
20C_8028	Analog USB2 480MHz PLL Control Register (CCM_ANALOG_PLL_USB2_CLR)	32	R/W	0001_2000h	<a href="#">15.7.3/585</a>
20C_802C	Analog USB2 480MHz PLL Control Register (CCM_ANALOG_PLL_USB2_TOG)	32	R/W	0001_2000h	<a href="#">15.7.3/585</a>
20C_8030	Analog System PLL Control Register (CCM_ANALOG_PLL_SYS)	32	R/W	0001_3001h	<a href="#">15.7.4/587</a>
20C_8034	Analog System PLL Control Register (CCM_ANALOG_PLL_SYS_SET)	32	R/W	0001_3001h	<a href="#">15.7.4/587</a>
20C_8038	Analog System PLL Control Register (CCM_ANALOG_PLL_SYS_CLR)	32	R/W	0001_3001h	<a href="#">15.7.4/587</a>
20C_803C	Analog System PLL Control Register (CCM_ANALOG_PLL_SYS_TOG)	32	R/W	0001_3001h	<a href="#">15.7.4/587</a>
20C_8070	Analog Audio PLL control Register (CCM_ANALOG_PLL_AUDIO)	32	R/W	0001_1006h	<a href="#">15.7.5/589</a>
20C_8074	Analog Audio PLL control Register (CCM_ANALOG_PLL_AUDIO_SET)	32	R/W	0001_1006h	<a href="#">15.7.5/589</a>
20C_8078	Analog Audio PLL control Register (CCM_ANALOG_PLL_AUDIO_CLR)	32	R/W	0001_1006h	<a href="#">15.7.5/589</a>
20C_807C	Analog Audio PLL control Register (CCM_ANALOG_PLL_AUDIO_TOG)	32	R/W	0001_1006h	<a href="#">15.7.5/589</a>
20C_8080	Numerator of Audio PLL Fractional Loop Divider Register (CCM_ANALOG_PLL_AUDIO_NUM)	32	R/W	05F5_E100h	<a href="#">15.7.6/591</a>
20C_8090	Denominator of Audio PLL Fractional Loop Divider Register (CCM_ANALOG_PLL_AUDIO_DENOM)	32	R/W	2964_619Ch	<a href="#">15.7.7/592</a>
20C_80A0	Analog Video PLL control Register (CCM_ANALOG_PLL_VIDEO)	32	R/W	0001_100Ch	<a href="#">15.7.8/593</a>
20C_80A4	Analog Video PLL control Register (CCM_ANALOG_PLL_VIDEO_SET)	32	R/W	0001_100Ch	<a href="#">15.7.8/593</a>
20C_80A8	Analog Video PLL control Register (CCM_ANALOG_PLL_VIDEO_CLR)	32	R/W	0001_100Ch	<a href="#">15.7.8/593</a>

*Table continues on the next page...*

**CCM\_ANALOG memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20C_80AC	Analog Video PLL control Register (CCM_ANALOG_PLL_VIDEO_TOG)	32	R/W	0001_100Ch	<a href="#">15.7.8/593</a>
20C_80B0	Numerator of Video PLL Fractional Loop Divider Register (CCM_ANALOG_PLL_VIDEO_NUM)	32	R/W	05F5_E100h	<a href="#">15.7.9/595</a>
20C_80C0	Denominator of Video PLL Fractional Loop Divider Register (CCM_ANALOG_PLL_VIDEO_DENOM)	32	R/W	10A2_4447h	<a href="#">15.7.10/596</a>
20C_80E0	Analog ENET PLL Control Register (CCM_ANALOG_PLL_ENET)	32	R/W	0001_1001h	<a href="#">15.7.11/597</a>
20C_80E4	Analog ENET PLL Control Register (CCM_ANALOG_PLL_ENET_SET)	32	R/W	0001_1001h	<a href="#">15.7.11/597</a>
20C_80E8	Analog ENET PLL Control Register (CCM_ANALOG_PLL_ENET_CLR)	32	R/W	0001_1001h	<a href="#">15.7.11/597</a>
20C_80EC	Analog ENET PLL Control Register (CCM_ANALOG_PLL_ENET_TOG)	32	R/W	0001_1001h	<a href="#">15.7.11/597</a>
20C_80F0	480MHz Clock (from PLL_USB2) Phase Fractional Divider Control Register (CCM_ANALOG_PFD_480)	32	R/W	1311_100Ch	<a href="#">15.7.12/599</a>
20C_80F4	480MHz Clock (from PLL_USB2) Phase Fractional Divider Control Register (CCM_ANALOG_PFD_480_SET)	32	R/W	1311_100Ch	<a href="#">15.7.12/599</a>
20C_80F8	480MHz Clock (from PLL_USB2) Phase Fractional Divider Control Register (CCM_ANALOG_PFD_480_CLR)	32	R/W	1311_100Ch	<a href="#">15.7.12/599</a>
20C_80FC	480MHz Clock (from PLL_USB2) Phase Fractional Divider Control Register (CCM_ANALOG_PFD_480_TOG)	32	R/W	1311_100Ch	<a href="#">15.7.12/599</a>
20C_8100	528MHz Clock (From PLL_SYS) Phase Fractional Divider Control Register (CCM_ANALOG_PFD_528)	32	R/W	1018_101Bh	<a href="#">15.7.13/601</a>
20C_8104	528MHz Clock (From PLL_SYS) Phase Fractional Divider Control Register (CCM_ANALOG_PFD_528_SET)	32	R/W	1018_101Bh	<a href="#">15.7.13/601</a>
20C_8108	528MHz Clock (From PLL_SYS) Phase Fractional Divider Control Register (CCM_ANALOG_PFD_528_CLR)	32	R/W	1018_101Bh	<a href="#">15.7.13/601</a>
20C_810C	528MHz Clock (From PLL_SYS) Phase Fractional Divider Control Register (CCM_ANALOG_PFD_528_TOG)	32	R/W	1018_101Bh	<a href="#">15.7.13/601</a>
20C_8150	Miscellaneous Control Register (CCM_ANALOG_MISC0)	32	R/W	0200_0000h	<a href="#">15.7.14/603</a>
20C_8154	Miscellaneous Control Register (CCM_ANALOG_MISC0_SET)	32	R/W	0200_0000h	<a href="#">15.7.14/603</a>
20C_8158	Miscellaneous Control Register (CCM_ANALOG_MISC0_CLR)	32	R/W	0200_0000h	<a href="#">15.7.14/603</a>
20C_815C	Miscellaneous Control Register (CCM_ANALOG_MISC0_TOG)	32	R/W	0200_0000h	<a href="#">15.7.14/603</a>
20C_8170	Miscellaneous Control Register (CCM_ANALOG_MISC2)	32	R/W	0027_2727h	<a href="#">15.7.15/604</a>
20C_8174	Miscellaneous Control Register (CCM_ANALOG_MISC2_SET)	32	R/W	0027_2727h	<a href="#">15.7.15/604</a>
20C_8178	Miscellaneous Control Register (CCM_ANALOG_MISC2_CLR)	32	R/W	0027_2727h	<a href="#">15.7.15/604</a>

*Table continues on the next page...*



**CCM\_ANALOG memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20C_817C	Miscellaneous Control Register (CCM_ANALOG_MISC2_TOG)	32	R/W	0027_2727h	<a href="#">15.7.15/604</a>

## 15.7.1 Analog ARM PLL control Register (CCM\_ANALOG\_PLL\_ARMn)

The control register provides control for the system PLL.

Address: 20C\_8000h base + 0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16								
R	LOCK								-								PLL_SEL		LVDS_24MHZ_SEL		LVDS_SEL		BYPASS	
W																								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1							

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BYPASS_CLK_SRC		ENABLE	POWERDOWN	Reserved				DIV_SELECT							
W																
Reset	0	0	1	1	0	0	0	0	0	1	0	0	0	0	1	0

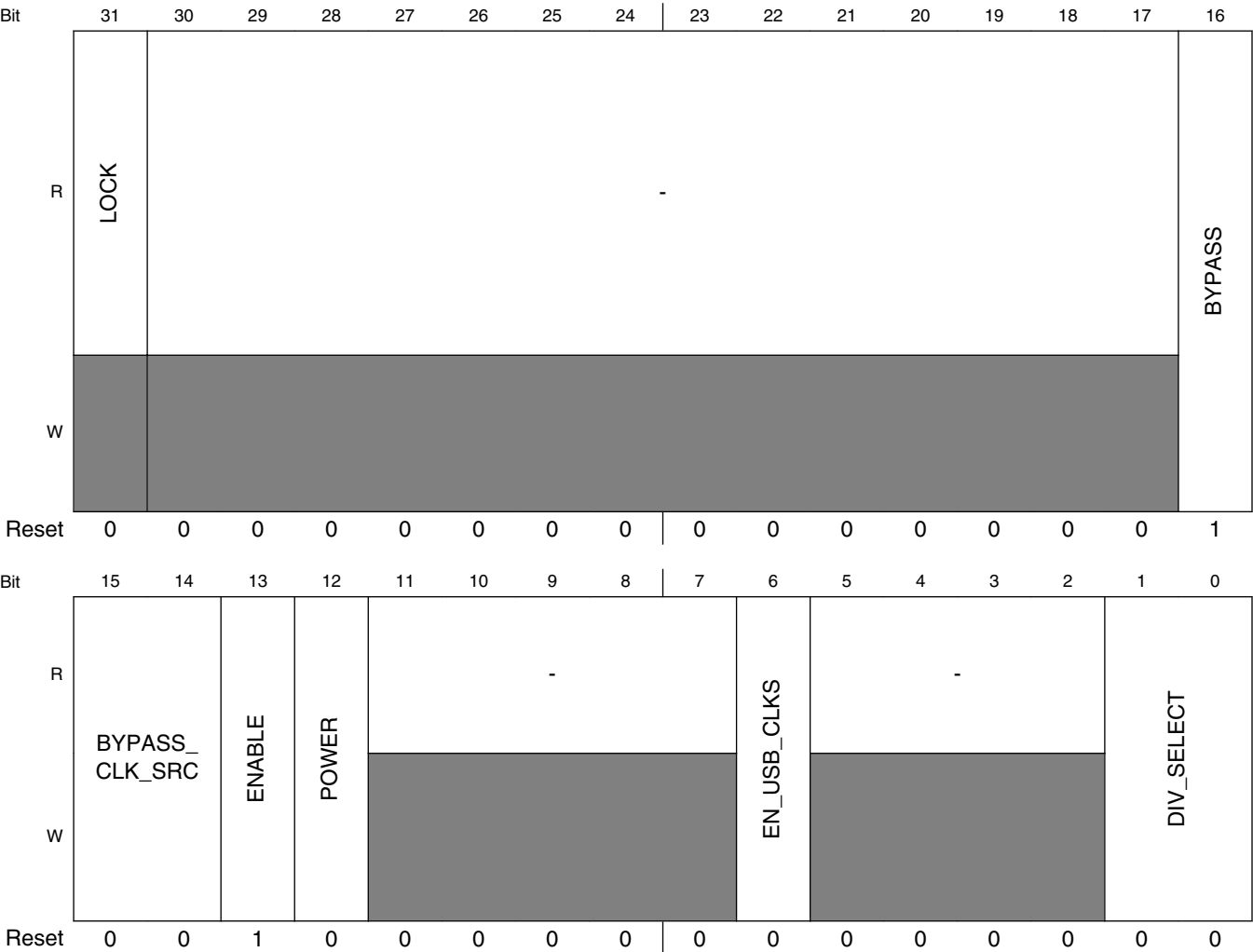
**CCM\_ANALOG\_PLL\_ARMn field descriptions**

Field	Description
31 LOCK	1 - PLL is currently locked. 0 - PLL is not currently locked.
30–20 -	Always set to zero (0).
19 PLL_SEL	Reserved
18 LVDS_24MHZ_SEL	Analog Debug Bit
17 LVDS_SEL	Analog Debug Bit
16 BYPASS	Bypass the pll.
15–14 BYPASS_CLK_SRC	Determines the bypass source. 0x0 <b>REF_CLK_24M</b> — Select the 24MHz oscillator as source. 0x1 <b>CLK1</b> — Select the CLK1_N / CLK1_P as source. 0x2 <b>Reserved</b> — 0x3 <b>Reserved</b> —
13 ENABLE	Enable the clock output.
12 POWERDOWN	Powers down the PLL.
11–7 -	This field is reserved. Reserved.
6–0 DIV_SELECT	This field controls the pll loop divider. Valid range for divider value: 54-108. $F_{out} = F_{in} * div\_select / 2.0$ .

### 15.7.2 Analog USB1 480MHz PLL Control Register (CCM\_ANALOG\_PLL\_USB1n)

The control register provides control for USBPHY0 480MHz PLL.

Address: 20C\_8000h base + 10h offset + (4d × i), where i=0d to 3d



CCM\_ANALOG\_PLL\_USB1n field descriptions

Field	Description
31 LOCK	1 - PLL is currently locked. 0 - PLL is not currently locked.
30–17 -	Always set to zero (0).
16 BYPASS	Bypass the pll.

Table continues on the next page...

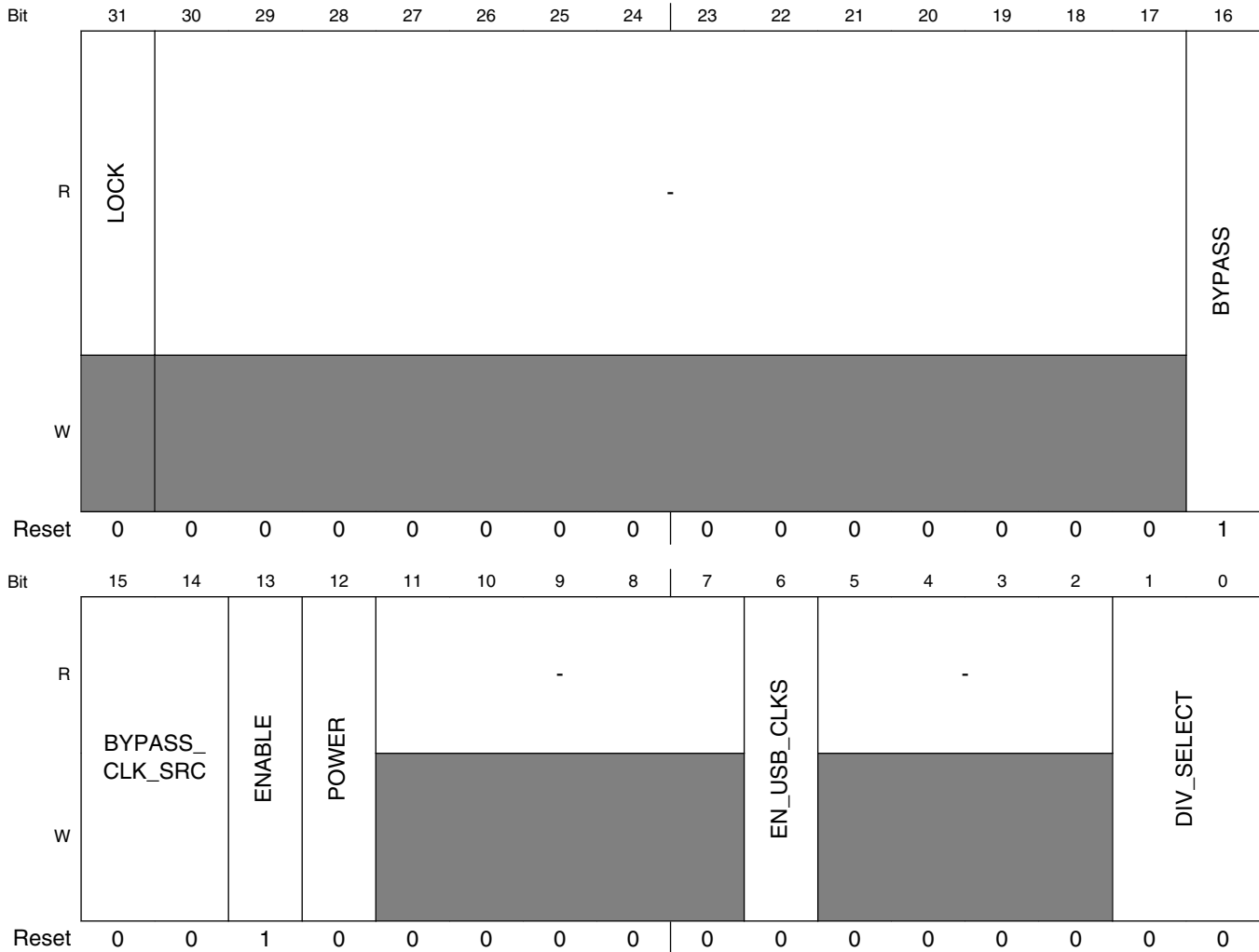
**CCM\_ANALOG\_PLL\_USB1<sub>n</sub> field descriptions (continued)**

Field	Description
15–14 BYPASS_CLK_SRC	Determines the bypass source. 0x0 <b>REF_CLK_24M</b> — Select the 24MHz oscillator as source. 0x1 <b>CLK1</b> — Select the CLK1_N / CLK1_P as source. 0x2 <b>GPANAIO</b> — 0x3 <b>CHRG_DET_B</b> —
13 ENABLE	Enable the PLL clock output.
12 POWER	Powers up the PLL. This bit will be set automatically when USBPHY0 remote wakeup event happens.
11–7 -	Always set to zero (0).
6 EN_USB_CLKS	Powers the 9-phase PLL outputs for USBPHY <sub>n</sub> . Additionally, the UTMI clock gate must be deasserted in the USBPHY <sub>n</sub> to enable USB <sub>n</sub> operation (clear CLKGATE bit in USBPHY <sub>n</sub> _CTRL). This bit will be set automatically when USBPHY <sub>n</sub> remote wakeup event occurs.  0 PLL outputs for USBPHY <sub>n</sub> off. 1 PLL outputs for USBPHY <sub>n</sub> on.
5–2 -	Always set to zero (0).
1–0 DIV_SELECT	This field controls the pll loop divider. 0 - Fout=Fref*20; 1 - Fout=Fref*22.

### 15.7.3 Analog USB2 480MHz PLL Control Register (CCM\_ANALOG\_PLL\_USB2n)

The control register provides control for USBPHY1 480MHz PLL.

Address: 20C\_8000h base + 20h offset + (4d × i), where i=0d to 3d



**CCM\_ANALOG\_PLL\_USB2n field descriptions**

Field	Description
31 LOCK	1 - PLL is currently locked. 0 - PLL is not currently locked.
30–17 -	Always set to zero (0).
16 BYPASS	Bypass the pll.

*Table continues on the next page...*

**CCM\_ANALOG\_PLL\_USB2n field descriptions (continued)**

Field	Description
15–14 BYPASS_CLK_SRC	Determines the bypass source. 0x0 <b>REF_CLK_24M</b> — Select the 24MHz oscillator as source. 0x1 <b>CLK1</b> — Select the CLK1_N / CLK1_P as source. 0x2 <b>Reserved</b> — 0x3 <b>Reserved</b> —
13 ENABLE	Enable the PLL clock output.
12 POWER	Powers up the PLL. This bit will be set automatically when USBPHY1 remote wakeup event happens.
11–7 -	Always set to zero (0).
6 EN_USB_CLKS	0: 8-phase PLL outputs for USBPHY1 are powered down. If set to 1, 8-phase PLL outputs for USBPHY1 are powered up. Additionally, the utmi clock gate must be deasserted in the USBPHY1 to enable USB0 operation (clear CLKGATE bit in USBPHY1_CTRL). This bit will be set automatically when USBPHY1 remote wakeup event happens.
5–2 -	Always set to zero (0).
1–0 DIV_SELECT	This field controls the pll loop divider. 0 - $F_{out}=F_{ref}*20$ ; 1 - $F_{out}=F_{ref}*22$ .

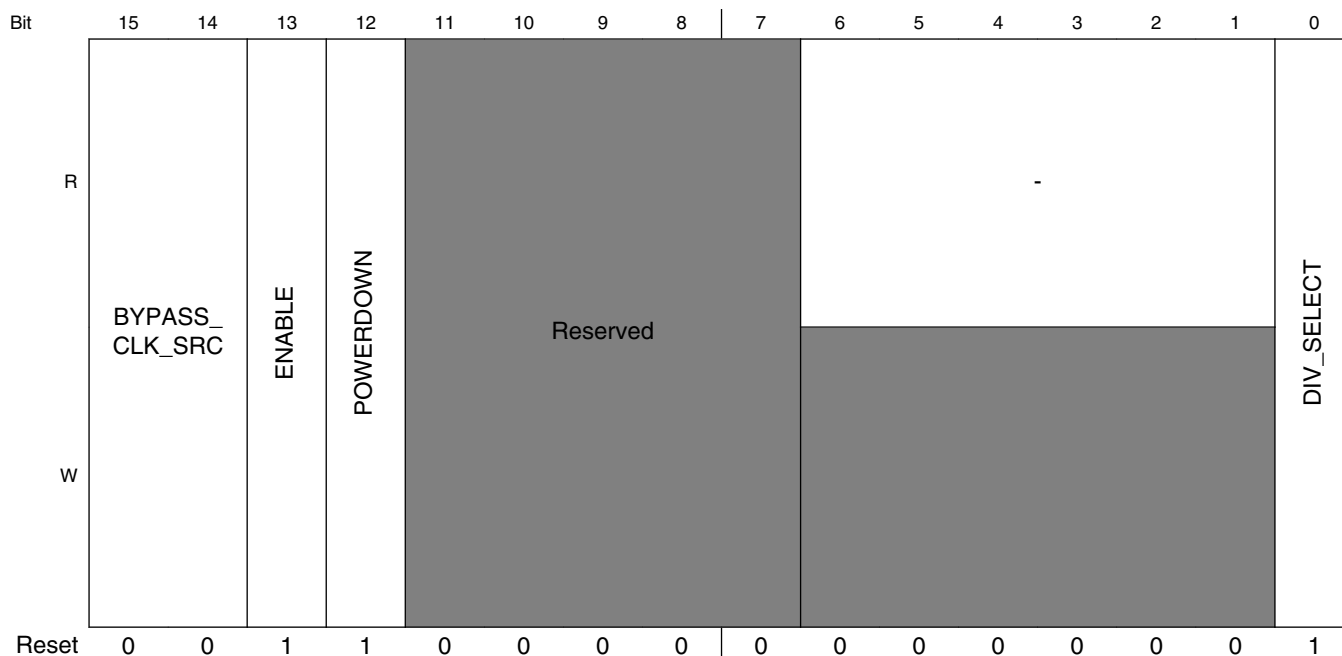
#### 15.7.4 Analog System PLL Control Register (CCM\_ANALOG\_PLL\_SYSn)

The control register provides control for the 528MHz PLL.

Address: 20C\_8000h base + 30h offset + (4d × i), where i=0d to 3d

Register 0: LOCK, PFD\_OFFSET\_EN, Reserved, BYPASS. Bit 31 is LOCK (R). Bits 18-17 are PFD\_OFFSET\_EN (R). Bits 16-15 are Reserved (R). Bit 14 is BYPASS (R).

## CCM Analog Memory Map/Register Definition



### CCM\_ANALOG\_PLL\_SYSn field descriptions

Field	Description
31 LOCK	1 - PLL is currently locked; 0 - PLL is not currently locked.
30–19 -	Always set to zero (0).
18 PFD_OFFSET_EN	Enables an offset in the phase frequency detector.
17 -	This field is reserved. Reserved
16 BYPASS	Bypass the pll.
15–14 BYPASS_CLK_SRC	Determines the bypass source. 0x0 <b>REF_CLK_24M</b> — Select the 24MHz oscillator as source. 0x1 <b>CLK1</b> — Select the CLK1_N / CLK1_P as source. 0x2 <b>GPANAIO</b> — 0x3 <b>CHRG_DET_B</b> —
13 ENABLE	Enable PLL output
12 POWERDOWN	Powers down the PLL.
11–7 -	This field is reserved. Reserved.
6–1 -	Always set to zero (0).
0 DIV_SELECT	This field controls the pll loop divider. 0 - Fout=Fref*20; 1 - Fout=Fref*22.



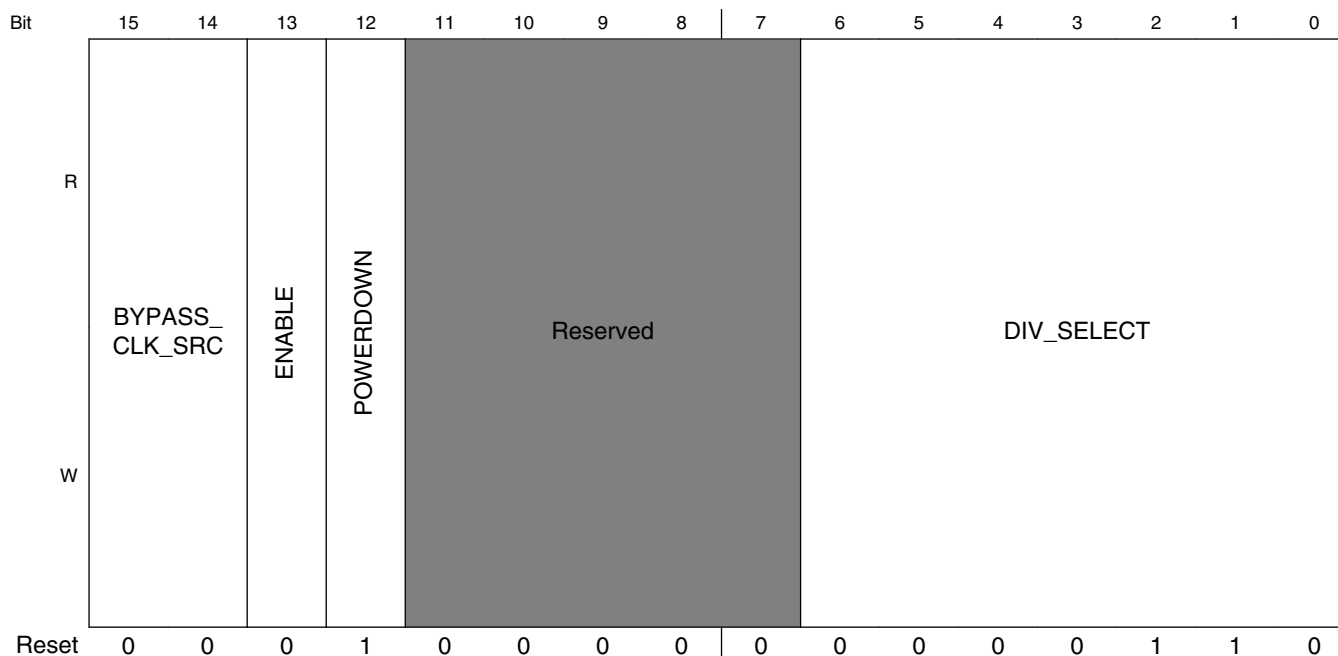
### 15.7.5 Analog Audio PLL control Register (CCM\_ANALOG\_PLL\_AUDION)

The control register provides control for the audio PLL.

Address: 20C\_8000h base + 70h offset + (4d × i), where i=0d to 3d

[illegible]

## CCM Analog Memory Map/Register Definition



### CCM\_ANALOG\_PLL\_AUDION field descriptions

Field	Description
31 LOCK	1 - PLL is currently locked. 0 - PLL is not currently locked.
30–22 -	Always set to zero (0).
21 SSC_EN	Reserved Bit
20–19 POST_DIV_SELECT	These bits implement a divider after the PLL, but before the enable and bypass mux. 00 — Divide by 4. 01 — Divide by 2. 10 — Divide by 1. 11 — Reserved
18 PFD_OFFSET_EN	Enables an offset in the phase frequency detector.
17 -	This field is reserved. Reserved
16 BYPASS	Bypass the pll.
15–14 BYPASS_CLK_SRC	Determines the bypass source. 0x0 <b>REF_CLK_24M</b> — Select the 24MHz oscillator as source. 0x1 <b>CLK1</b> — Select the CLK1_N / CLK1_P as source. 0x2 <b>Reserved</b> — 0x3 <b>Reserved</b> —

Table continues on the next page...

**CCM\_ANALOG\_PLL\_AUDIO<sub>n</sub> field descriptions (continued)**

Field	Description
13 ENABLE	Enable PLL output
12 POWERDOWN	Powers down the PLL.
11–7 -	This field is reserved. Reserved.
6–0 DIV_SELECT	This field controls the pll loop divider. Valid range for DIV_SELECT divider value: 27~54.

**15.7.6 Numerator of Audio PLL Fractional Loop Divider Register (CCM\_ANALOG\_PLL\_AUDIO\_NUM)**

This register contains the numerator (A) of Audio PLL fractional loop divider.(Signed number), absolute value should be less than denominator

Absolute value should be less than denominator

Address: 20C\_8000h base + 80h offset = 20C\_8080h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	-																															
W																																
Reset	0	0	0	0	0	0	1	0	1	1	1	1	0	1	0	1	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0

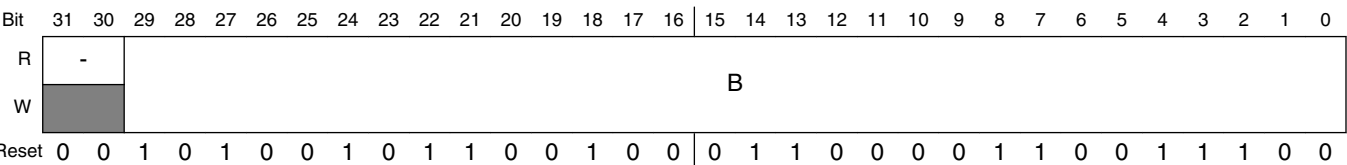
**CCM\_ANALOG\_PLL\_AUDIO\_NUM field descriptions**

Field	Description
31–30 -	Always set to zero (0).
29–0 A	30 bit numerator of fractional loop divider.

### 15.7.7 Denominator of Audio PLL Fractional Loop Divider Register (CCM\_ANALOG\_PLL\_AUDIO\_DENOM)

This register contains the Denominator (B) of Audio PLL fractional loop divider. (unsigned number)

Address: 20C\_8000h base + 90h offset = 20C\_8090h



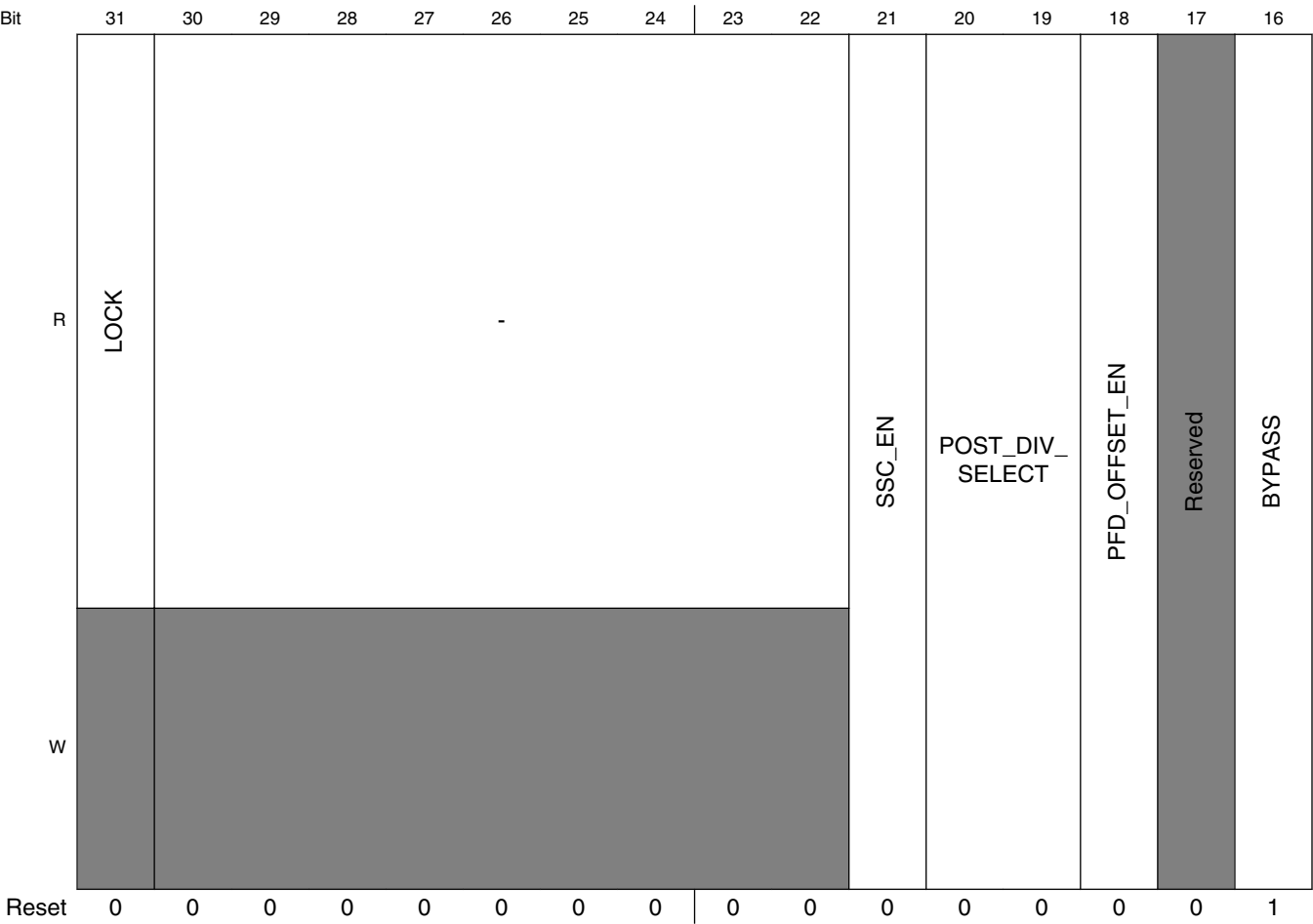
CCM\_ANALOG\_PLL\_AUDIO\_DENOM field descriptions

Field	Description
31–30 -	Always set to zero (0).
29–0 B	30 bit Denominator of fractional loop divider.

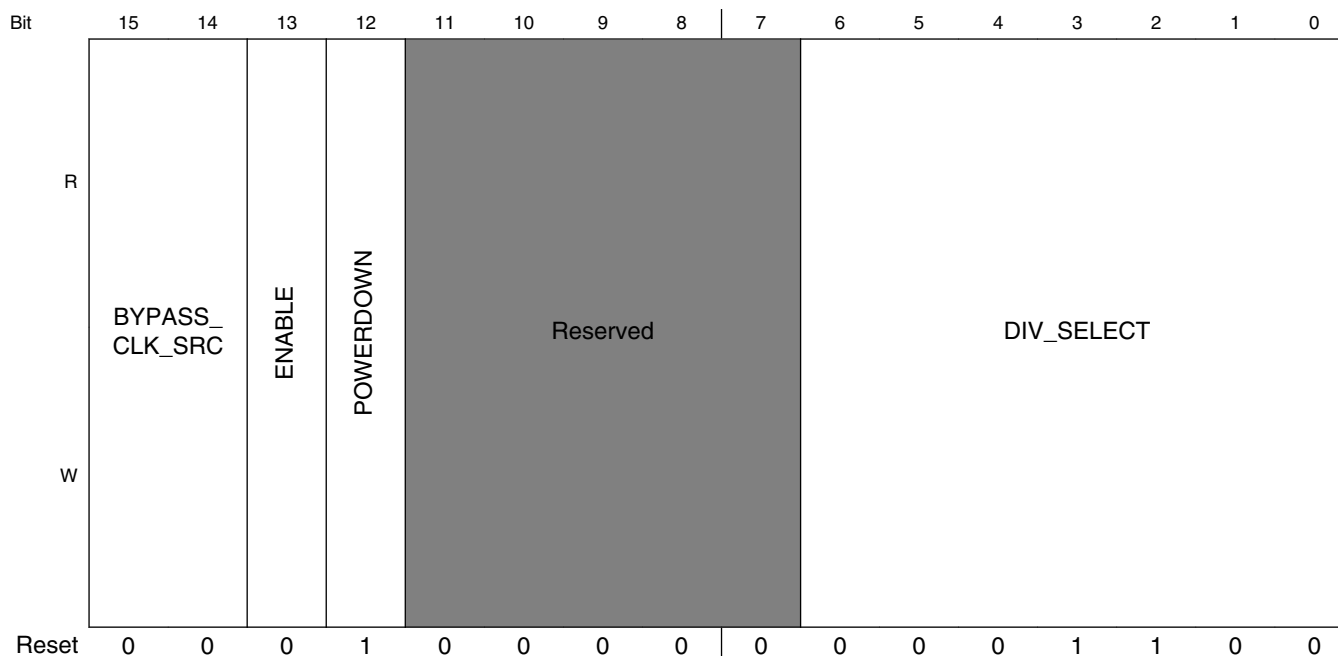
### 15.7.8 Analog Video PLL control Register (CCM\_ANALOG\_PLL\_VIDEO*n*)

The control register provides control for the Video PLL.

Address: 20C\_8000h base + A0h offset + (4d × i), where i=0d to 3d



## CCM Analog Memory Map/Register Definition



### CCM\_ANALOG\_PLL\_VIDEOn field descriptions

Field	Description
31 LOCK	1 - PLL is currently locked; 0 - PLL is not currently locked.
30–22 -	Always set to zero (0).
21 SSC_EN	Reserved Bit
20–19 POST_DIV_SELECT	These bits implement a divider after the PLL, but before the enable and bypass mux. 00 — Divide by 4. 01 — Divide by 2. 10 — Divide by 1. 11 — Reserved
18 PFD_OFFSET_EN	Enables an offset in the phase frequency detector.
17 -	This field is reserved. Reserved
16 BYPASS	Bypass the pll.
15–14 BYPASS_CLK_SRC	Determines the bypass source. 0x0 <b>REF_CLK_24M</b> — Select the 24MHz oscillator as source. 0x1 <b>CLK1</b> — Select the CLK1_N / CLK1_P as source. 0x2 <b>Reserved</b> — 0x3 <b>Reserved</b> —

Table continues on the next page...

**CCM\_ANALOG\_PLL\_VIDEO<sub>n</sub> field descriptions (continued)**

Field	Description
13 ENABLE	Enable PLL output
12 POWERDOWN	Powers down the PLL.
11–7 -	This field is reserved. Reserved.
6–0 DIV_SELECT	This field controls the pll loop divider. Valid range for DIV_SELECT divider value: 27~54.

**15.7.9 Numerator of Video PLL Fractional Loop Divider Register (CCM\_ANALOG\_PLL\_VIDEO\_NUM)**

This register contains the numerator (A) of Video PLL fractional loop divider.(Signed number)

Absolute value should be less than denominator

Address: 20C\_8000h base + B0h offset = 20C\_80B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	-																															
W																																
Reset	0	0	0	0	0	0	1	0	1	1	1	1	0	1	0	1	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0

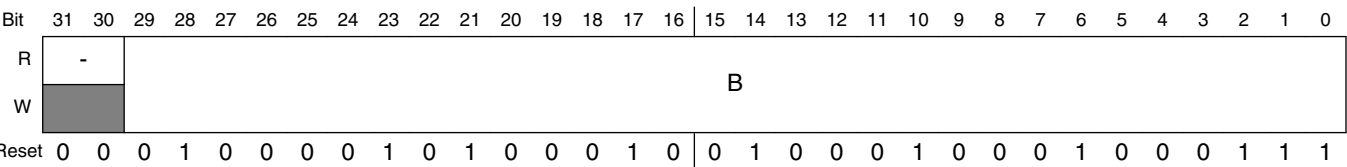
**CCM\_ANALOG\_PLL\_VIDEO\_NUM field descriptions**

Field	Description
31–30 -	Always set to zero (0).
29–0 A	30 bit numerator of fractional loop divider(Signed number), absolute value should be less than denominator

### 15.7.10 Denominator of Video PLL Fractional Loop Divider Register (CCM\_ANALOG\_PLL\_VIDEO\_DENOM)

This register contains the Denominator (B) of Video PLL fractional loop divider.  
(Unsigned number)

Address: 20C\_8000h base + C0h offset = 20C\_80C0h



CCM\_ANALOG\_PLL\_VIDEO\_DENOM field descriptions

Field	Description
31–30 -	Always set to zero (0).
29–0 B	30 bit Denominator of fractional loop divider.



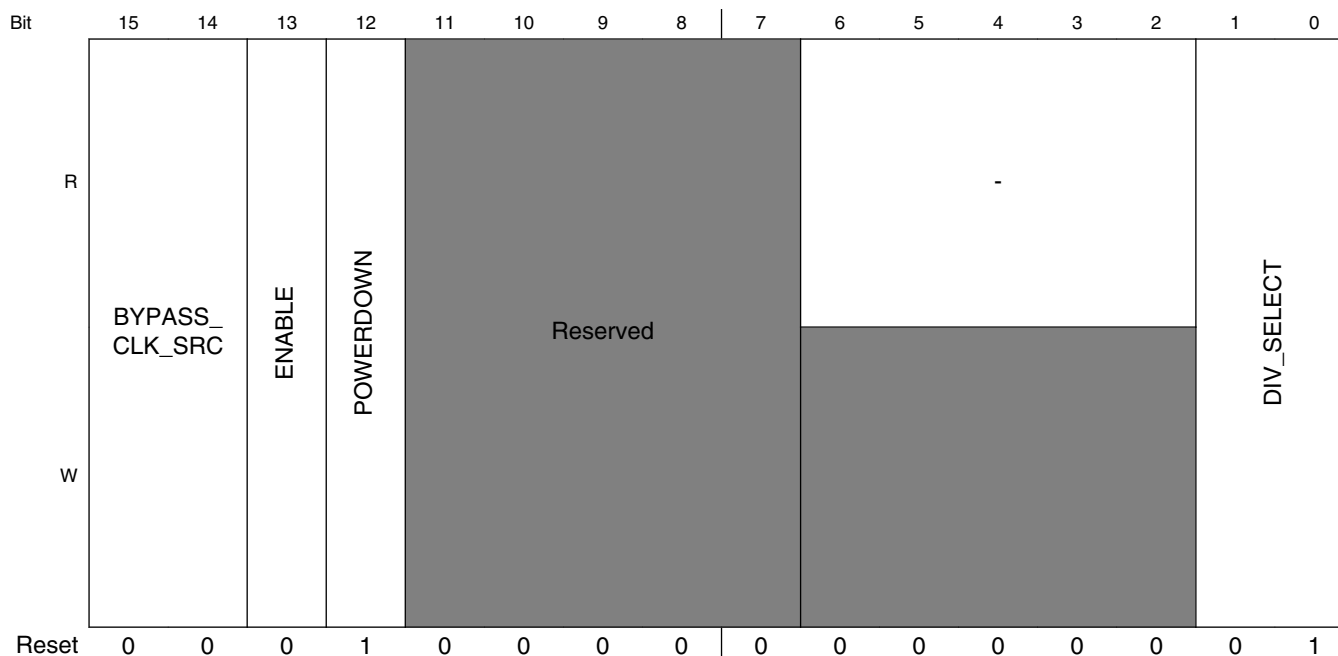
### 15.7.11 Analog ENET PLL Control Register (CCM\_ANALOG\_PLL\_ENETn)

The control register provides control for the ENET PLL.

Address: 20C\_8000h base + E0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LOCK											ENABLE_100M	ENABLE_125M	PFD_OFFSET_EN	Reserved	BYPASS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

## CCM Analog Memory Map/Register Definition



### CCM\_ANALOG\_PLL\_ENETn field descriptions

Field	Description
31 LOCK	1 - PLL is currently locked; 0 - PLL is not currently locked.
30–21 -	Always set to zero (0).
20 ENABLE_100M	Enables an offset in the phase frequency detector.
19 ENABLE_125M	Enables an offset in the phase frequency detector.
18 PFD_OFFSET_EN	Enables an offset in the phase frequency detector.
17 -	This field is reserved. Reserved
16 BYPASS	Bypass the pll.
15–14 BYPASS_CLK_SRC	Determines the bypass source. 0x0 <b>REF_CLK_24M</b> — Select the 24MHz oscillator as source. 0x1 <b>CLK1</b> — Select the CLK1_N / CLK1_P as source. 0x2 <b>Reserved</b> — 0x3 <b>Reserved</b> —
13 ENABLE	Enable the ethernet clock output.
12 POWERDOWN	Powers down the PLL.
11–7 -	This field is reserved. Reserved.

Table continues on the next page...

**CCM\_ANALOG\_PLL\_ENET<sub>n</sub> field descriptions (continued)**

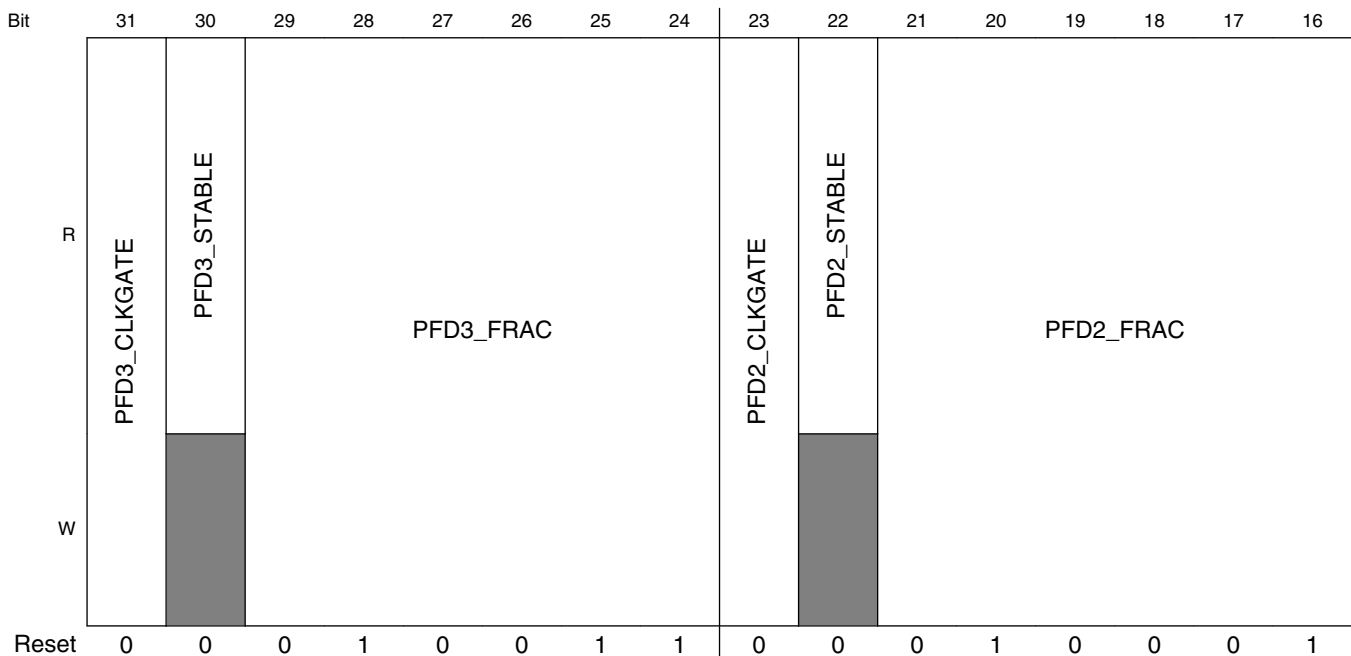
Field	Description
6–2 -	Always set to zero (0).
1–0 DIV_SELECT	Controls the frequency of the ethernet reference clock. 00 - 25MHz; 01 - 50MHz; 10 - 100MHz (not 50% duty cycle); 11 - 125MHz;

### 15.7.12 480MHz Clock (from PLL\_USB2) Phase Fractional Divider Control Register (CCM\_ANALOG\_PFD\_480<sub>n</sub>)

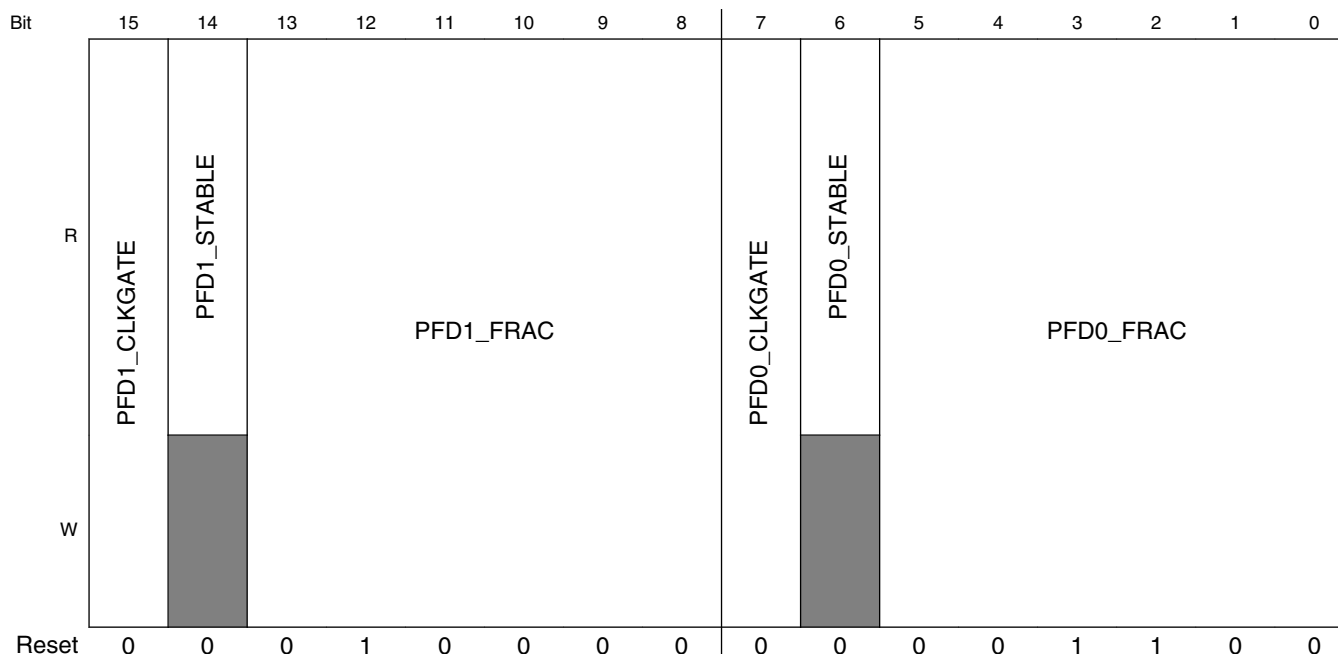
The PFD\_480 control register provides control for PFD clock generation.

This register controls the 4-phase fractional clock dividers. The fractional clock frequencies are a product of the values in these registers.

Address: 20C\_8000h base + F0h offset + (4d × i), where i=0d to 3d



## CCM Analog Memory Map/Register Definition



### CCM\_ANALOG\_PFD\_480n field descriptions

Field	Description
31 PFD3_CLKGATE	IO Clock Gate. If set to 1, the 3rd fractional divider clock (reference ref_pfd3) is off (power savings). 0: ref_pfd3 fractional divider clock is enabled. Need to assert this bit before PLL is powered down
30 PFD3_STABLE	This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit, program the new value, and when this bit inverts, the phase divider clock output is stable. Note that the value will not invert when the fractional divider is taken out of or placed into clock-gated state.
29–24 PFD3_FRAC	This field controls the fractional divide value. The resulting frequency shall be $480 \times 18 / \text{PFD3\_FRAC}$ where PFD3_FRAC is in the range 12-35.
23 PFD2_CLKGATE	IO Clock Gate. If set to 1, the IO fractional divider clock (reference ref_pfd2) is off (power savings). 0: ref_pfd2 fractional divider clock is enabled. Need to assert this bit before PLL is powered down
22 PFD2_STABLE	This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit, program the new value, and when this bit inverts, the phase divider clock output is stable. Note that the value will not invert when the fractional divider is taken out of or placed into clock-gated state.
21–16 PFD2_FRAC	This field controls the fractional divide value. The resulting frequency shall be $480 \times 18 / \text{PFD2\_FRAC}$ where PFD2_FRAC is in the range 12-35.
15 PFD1_CLKGATE	IO Clock Gate. If set to 1, the IO fractional divider clock (reference ref_pfd1) is off (power savings). 0: ref_pfd1 fractional divider clock is enabled. Need to assert this bit before PLL is powered down
14 PFD1_STABLE	This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit,

Table continues on the next page...

**CCM\_ANALOG\_PFD\_480n field descriptions (continued)**

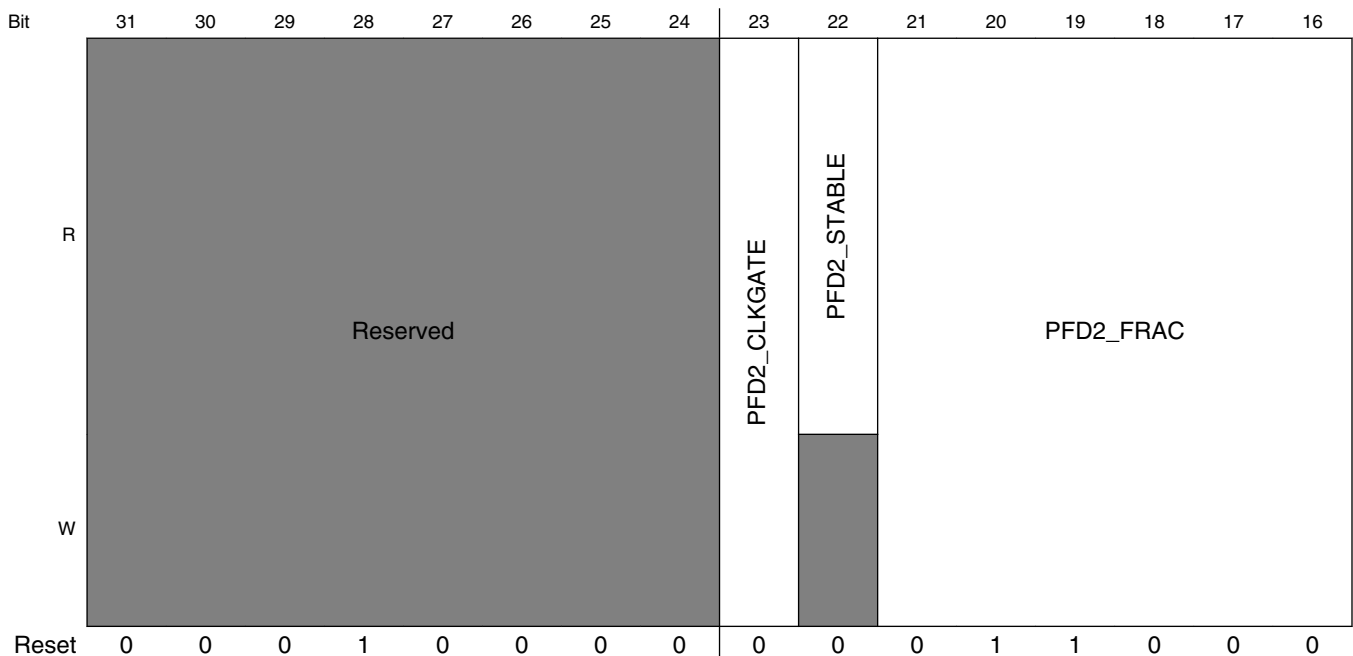
Field	Description
	program the new value, and when this bit inverts, the phase divider clock output is stable. Note that the value will not invert when the fractional divider is taken out of or placed into clock-gated state.
13–8 PFD1_FRAC	This field controls the fractional divide value. The resulting frequency shall be $480 \times 18 / \text{PFD1\_FRAC}$ where PFD1_FRAC is in the range 12–35.
7 PFD0_CLKGATE	If set to 1, the IO fractional divider clock (reference ref_pfd0) is off (power savings). 0: ref_pfd0 fractional divider clock is enabled.  Need to assert this bit before PLL is powered down
6 PFD0_STABLE	This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit, program the new value, and when this bit inverts, the phase divider clock output is stable. Note that the value will not invert when the fractional divider is taken out of or placed into clock-gated state.
5–0 PFD0_FRAC	This field controls the fractional divide value. The resulting frequency shall be $480 \times 18 / \text{PFD0\_FRAC}$ where PFD0_FRAC is in the range 12–35.

**15.7.13 528MHz Clock (From PLL\_SYS) Phase Fractional Divider Control Register (CCM\_ANALOG\_PFD\_528n)**

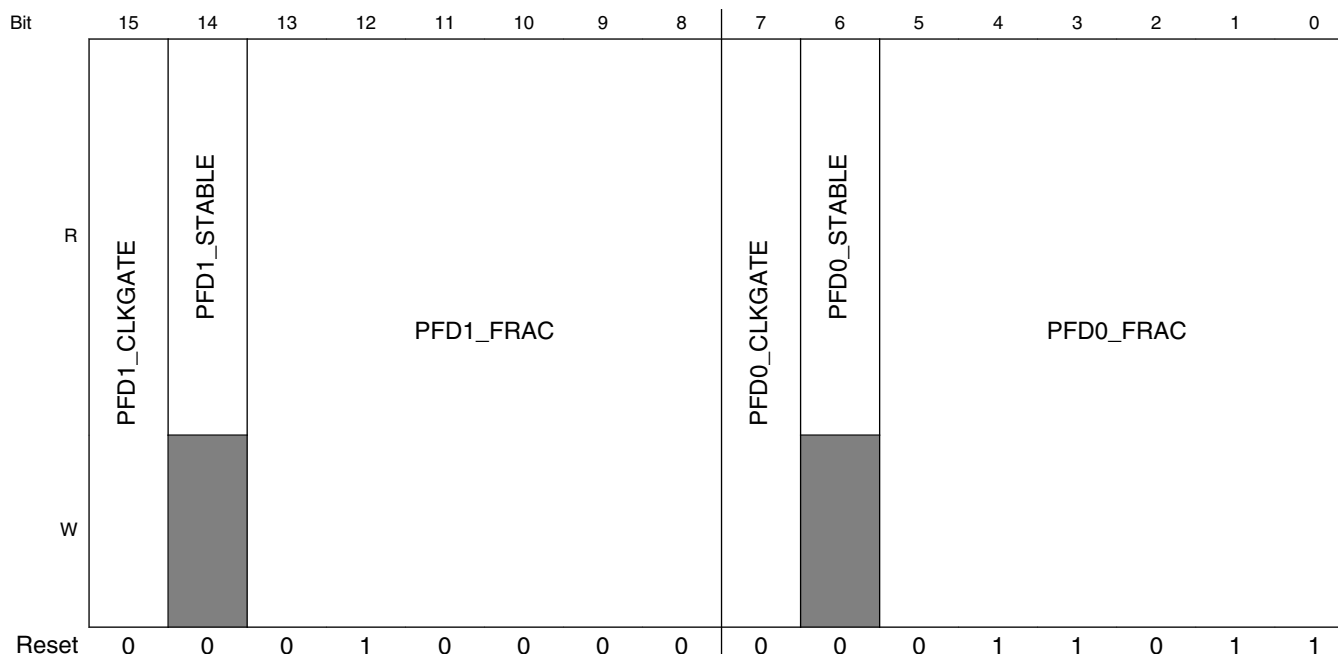
The PFD\_528 control register provides control for PFD clock generation.

This register controls the 3-phase fractional clock dividers. The fractional clock frequencies are a product of the values in these registers.

Address:  $20C\_8000h \text{ base} + 100h \text{ offset} + (4d \times i)$ , where  $i=0d \text{ to } 3d$



## CCM Analog Memory Map/Register Definition



### CCM\_ANALOG\_PFD\_528n field descriptions

Field	Description
31-24 -	This field is reserved. Reserved
23 PFD2_CLKGATE	IO Clock Gate. If set to 1, the IO fractional divider clock (reference ref_pfd2) is off (power savings). 0: ref_pfd2 fractional divider clock is enabled. Need to assert this bit before PLL powered down
22 PFD2_STABLE	This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit, program the new value, and when this bit inverts, the phase divider clock output is stable. Note that the value will not invert when the fractional divider is taken out of or placed into clock-gated state.
21-16 PFD2_FRAC	This field controls the fractional divide value. The resulting frequency shall be $528 \times 18 / \text{PFD2\_FRAC}$ where PFD2_FRAC is in the range 12-35.
15 PFD1_CLKGATE	IO Clock Gate. If set to 1, the IO fractional divider clock (reference ref_pfd1) is off (power savings). 0: ref_pfd1 fractional divider clock is enabled. Need to assert this bit before PLL powered down
14 PFD1_STABLE	This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit, program the new value, and when this bit inverts, the phase divider clock output is stable. Note that the value will not invert when the fractional divider is taken out of or placed into clock-gated state.
13-8 PFD1_FRAC	This field controls the fractional divide value. The resulting frequency shall be $528 \times 18 / \text{PFD1\_FRAC}$ where PFD1_FRAC is in the range 12-35.
7 PFD0_CLKGATE	If set to 1, the IO fractional divider clock (reference ref_pfd0) is off (power savings). 0: ref_pfd0 fractional divider clock is enabled. Need to assert this bit before PLL powered down

Table continues on the next page...

**CCM\_ANALOG\_PFD\_528n field descriptions (continued)**

Field	Description
6 PFD0_STABLE	This read-only bitfield is for DIAGNOSTIC PURPOSES ONLY since the fractional divider should become stable quickly enough that this field will never need to be used by either device driver or application code. The value inverts when the new programmed fractional divide value has taken effect. Read this bit, program the new value, and when this bit inverts, the phase divider clock output is stable. Note that the value will not invert when the fractional divider is taken out of or placed into clock-gated state.
5–0 PFD0_FRAC	This field controls the fractional divide value. The resulting frequency shall be $528 \times 18 / \text{PFD0\_FRAC}$ where PFD0_FRAC is in the range 12–35.

**15.7.14 Miscellaneous Control Register (CCM\_ANALOG\_MISC0n)**

This register defines the control for miscellaneous CCM Analog blocks.

Address: 20C\_8000h base + 150h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved			STOP_MODE_CONFIG		Reserved										
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**CCM\_ANALOG\_MISC0n field descriptions**

Field	Description
31–13 -	This field is reserved. Reserved
12–11 STOP_MODE_CONFIG	Configure the analog behavior in stop mode. 00 All the analog domain except the RTC is powered down on STOP mode assertion 01 All the analog domain except the LDO_1P1 and LDO_2P5 regulators are powered down on STOP mode assertion. If required the CCM can be configured to not power down the oscillator (XTALOSC) 10 Reserved 11 Reserved
10–0 -	This field is reserved. Reserved

## 15.7.15 Miscellaneous Control Register (CCM\_ANALOG\_MISC2n)

This register defines the control for miscellaneous CCM Analog blocks.

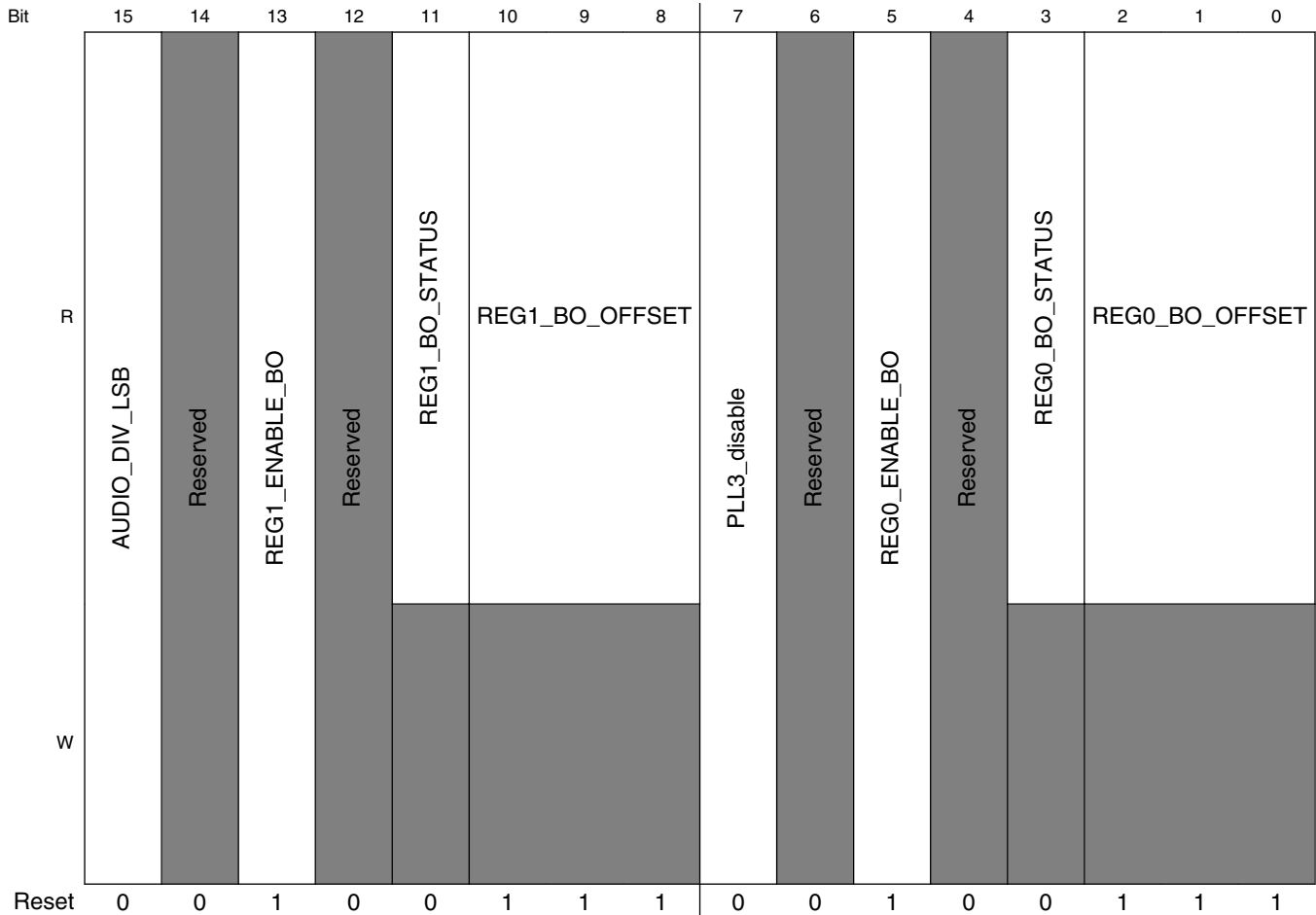
### NOTE

This register is shared with PMU.

Address: 20C\_8000h base + 170h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1





CCM\_ANALOG\_MISC2n field descriptions

Field	Description
31–30 VIDEO_DIV	<p>Post-divider for video. The output clock of the video PLL should be gated prior to changing this divider to prevent glitches. This divider is feed by PLL_VIDEOOn[POST_DIV_SELECT] to achieve division ratios of / 1, /2, /4, /8, and /16.</p> <p>00 divide by 1 (Default)            01 divide by 2            10 divide by 1            11 divide by 4</p>
29–28 REG2_STEP_TIME	<p>Number of clock periods (24MHz clock).</p> <p><b>NOTE:</b> Not related to CCM. See <a href="#">Power Management Unit (PMU)</a> for more information.</p> <p>00 <b>64_CLOCKS</b> — 64            01 <b>128_CLOCKS</b> — 128            10 <b>256_CLOCKS</b> — 256            11 <b>512_CLOCKS</b> — 512</p>
27–26 REG1_STEP_TIME	<p>Number of clock periods (24MHz clock).</p> <p><b>NOTE:</b> Not related to CCM. See <a href="#">Power Management Unit (PMU)</a> for more information.</p>

Table continues on the next page...

## CCM\_ANALOG\_MISC2n field descriptions (continued)

Field	Description
	00 <b>64_CLOCKS</b> — 64 01 <b>128_CLOCKS</b> — 128 10 <b>256_CLOCKS</b> — 256 11 <b>512_CLOCKS</b> — 512
25–24 REG0_STEP_ TIME	Number of clock periods (24MHz clock).  <b>NOTE:</b> Not related to CCM. See <a href="#">Power Management Unit (PMU)</a> for more information.  00 <b>64_CLOCKS</b> — 64 01 <b>128_CLOCKS</b> — 128 10 <b>256_CLOCKS</b> — 256 11 <b>512_CLOCKS</b> — 512
23 AUDIO_DIV_ MSB	MSB of Post-divider for Audio PLL. The output clock of the video PLL should be gated prior to changing this divider to prevent glitches. This divider is feed by PLL_AUDION[POST_DIV_SELECT] to achieve division ratios of /1, /2, /4, /8, and /16.  <b>NOTE:</b> MSB bit value pertains to the first bit, please program the LSB bit (bit 15) as well to change divider value  00 divide by 1 (Default) 01 divide by 2 10 divide by 1 11 divide by 4
22 REG2_OK	Signals that the voltage is above the brownout level for the SOC supply. 1 = regulator output > brownout_target  <b>NOTE:</b> Not related to CCM. See <a href="#">Power Management Unit (PMU)</a> for more information.
21 REG2_ENABLE_ BO	Enables the brownout detection.  <b>NOTE:</b> Not related to CCM. See <a href="#">Power Management Unit (PMU)</a> for more information.
20 -	This field is reserved.
19 REG2_BO_ STATUS	Reg2 brownout status bit.  <b>NOTE:</b> Not related to CCM. See <a href="#">Power Management Unit (PMU)</a> for more information.
18–16 REG2_BO_ OFFSET	This field defines the brown out voltage offset for the xPU power domain. IRQ_DIG_BO is also asserted. Single-bit increments reflect 25mV brownout voltage steps. The reset brown-offset is 175mV below the programmed target code. Brownout target = OUTPUT_TRG - BO_OFFSET. Some steps may be irrelevant because of input supply limitations or load operation.  <b>NOTE:</b> Not related to CCM. See <a href="#">Power Management Unit (PMU)</a> for more information.  100 Brownout offset = 0.100V 111 Brownout offset = 0.175V
15 AUDIO_DIV_LSB	LSB of Post-divider for Audio PLL. The output clock of the video PLL should be gated prior to changing this divider to prevent glitches. This divider is feed by PLL_AUDION[POST_DIV_SELECT] to achieve division ratios of /1, /2, /4, /8, and /16.  <b>NOTE:</b> LSB bit value pertains to the last bit, please program the MSB bit (bit 23) as well, to change divider value  00 divide by 1 (Default)

Table continues on the next page...

**CCM\_ANALOG\_MISC2n field descriptions (continued)**

Field	Description
	01 divide by 2 10 divide by 1 11 divide by 4
14 -	This field is reserved. Reserved
13 REG1_ENABLE_ BO	Enables the brownout detection. <b>NOTE:</b> Not related to CCM. See <a href="#">Power Management Unit (PMU)</a> for more information.
12 -	This field is reserved.
11 REG1_BO_ STATUS	Reg1 brownout status bit. <b>NOTE:</b> Not related to CCM. See <a href="#">Power Management Unit (PMU)</a> for more information. 1 Brownout, supply is below target minus brownout offset.
10–8 REG1_BO_ OFFSET	This field defines the brown out voltage offset for the xPU power domain. IRQ_DIG_BO is also asserted. Single-bit increments reflect 25mV brownout voltage steps. The reset brown-offset is 175mV below the programmed target code. Brownout target = OUTPUT_TRG - BO_OFFSET. Some steps may be irrelevant because of input supply limitations or load operation. <b>NOTE:</b> Not related to CCM. See <a href="#">Power Management Unit (PMU)</a> for more information. 100 Brownout offset = 0.100V 111 Brownout offset = 0.175V
7 PLL3_disable	Default value of "0". Should be set to "1" to turn off the USB-PLL(PLL3) in run mode.
6 -	This field is reserved.
5 REG0_ENABLE_ BO	Enables the brownout detection. <b>NOTE:</b> Not related to CCM. See <a href="#">Power Management Unit (PMU)</a> for more information.
4 -	This field is reserved.
3 REG0_BO_ STATUS	Reg0 brownout status bit. <b>NOTE:</b> Not related to CCM. See <a href="#">Power Management Unit (PMU)</a> for more information. 1 Brownout, supply is below target minus brownout offset.
2–0 REG0_BO_ OFFSET	This field defines the brown out voltage offset for the CORE power domain. IRQ_DIG_BO is also asserted. Single-bit increments reflect 25mV brownout voltage steps. Some steps may be irrelevant because of input supply limitations or load operation. <b>NOTE:</b> Not related to CCM. See <a href="#">Power Management Unit (PMU)</a> for more information. 100 Brownout offset = 0.100V 111 Brownout offset = 0.175V



# Chapter 16

## CMOS Sensor Interface (CSI)

### 16.1 Overview

This chapter presents the CMOS Sensor Interface (CSI) architecture, operation principles, and programming model.

The CSI enables the chip to connect directly to external CMOS image sensors. CMOS image sensors are separated into two classes, dumb and smart. Dumb sensors are those that support only traditional sensor timing (Vertical SYNC and Horizontal SYNC) and output only Bayer and statistics data, while smart sensors support CCIR656 video decoder formats and perform additional processing of the image (for example, image compression, image pre-filtering, and various data output formats).

The capabilities of the CSI include:

- Configurable interface logic to support most commonly available CMOS sensors.
- Support for CCIR656 video interface as well as traditional sensor interface.
- 8-bit data port for YCbCr, YUV, or RGB data input.
- 8-bit/10-bit/16-bit data port for Bayer data input.
- Full control of 8-bit/pixel, 10-bit/pixel or 16-bit/pixel data format to 32-bit receive FIFO packing.
- 128 x 32 FIFO to store received image pixel data.
- Receive FIFO overrun protection mechanism.
- Embedded DMA controllers to transfer data from receive FIFO or statistic FIFO through AHB bus.
- Support 2D DMA transfer from the receive FIFO to the frame buffers in the external memory.
- Support double buffering two frames in the external memory.
- Single interrupt source to interrupt controller from maskable interrupt sources: Start of Frame, End of Frame, Change of Field, FIFO full, FIFO overrun, DMA transfer done, CCIR error and AHB bus response error.

- Configurable master clock frequency output to sensor.
- Statistic data generation for Auto Exposure (AE) and Auto White Balance (AWB) control of the camera (only for Bayer data and 8-bit/pixel format).

## 16.2 External Signals

The table below describes the external signals for the CSI. The external signals are tied between the CSI module and an external CMOS sensor.

**Table 16-1. CSI External Signals**

Signal	Description	Pad	Mode	Direction
CSI_DATA00	Data Sensor Signal 0, part of 16-bit Sensor Data Bus (Bayer, YUV, YCrCb, RGB)	EPDC_D0	ALT3	I
		LCD_DAT17	ALT2	
		SD2_CLK	ALT3	
CSI_DATA01	Data Sensor Signal 1, part of 16-bit Sensor Data Bus (Bayer, YUV, YCrCb, RGB)	EPDC_D1	ALT3	I
		LCD_DAT16	ALT2	
		SD2_CMD	ALT3	
CSI_DATA02	Data Sensor Signal 2, part of 16-bit Sensor Data Bus (Bayer, YUV, YCrCb, RGB)	EPDC_D2	ALT3	I
		LCD_DAT15	ALT2	
		SD2_DAT0	ALT3	
CSI_DATA03	Data Sensor Signal 3, part of 16-bit Sensor Data Bus (Bayer, YUV, YCrCb, RGB)	EPDC_D3	ALT3	I
		LCD_DAT14	ALT2	
		SD2_DAT1	ALT3	
CSI_DATA04	Data Sensor Signal 4, part of 16-bit Sensor Data Bus (Bayer, YUV, YCrCb, RGB)	EPDC_D4	ALT3	I
		LCD_DAT13	ALT2	
		SD2_DAT2	ALT3	
CSI_DATA05	Data Sensor Signal 5, part of 16-bit Sensor Data Bus (Bayer, YUV, YCrCb, RGB)	EPDC_D5	ALT3	I
		LCD_DAT12	ALT2	
		SD2_DAT3	ALT3	
CSI_DATA06	Data Sensor Signal 6, part of 16-bit Sensor Data Bus (Bayer, YUV, YCrCb, RGB)	EPDC_D6	ALT3	I
		LCD_DAT11	ALT2	
		SD2_DAT4	ALT3	
CSI_DATA07	Data Sensor Signal 7, part of 16-bit Sensor Data Bus (Bayer, YUV, YCrCb, RGB)	EPDC_D7	ALT3	I
		LCD_DAT10	ALT2	
		SD2_DAT5	ALT3	
CSI_DATA08	Data Sensor Signal 8, part of 16-bit Sensor Data Bus (Bayer, YUV, YCrCb, RGB)	EPDC_SDCLK	ALT3	I
		LCD_DAT9	ALT2	
		SD2_DAT6	ALT3	

*Table continues on the next page...*

**Table 16-1. CSI External Signals (continued)**

Signal	Description	Pad	Mode	Direction
CSI_DATA09	Data Sensor Signal 9, part of 16-bit Sensor Data Bus (Bayer, YUV, YCrCb, RGB)	EPDC_SDLE	ALT3	I
		LCD_DAT8	ALT2	
		SD2_DAT7	ALT3	
CSI_DATA10	Data Sensor Signal 10, part of 16-bit Sensor Data Bus (Bayer, YUV, YCrCb, RGB)	EPDC_SDOE	ALT3	I
		LCD_DAT23	ALT2	
		SD3_CLK	ALT3	
CSI_DATA11	Data Sensor Signal 11, part of 16-bit Sensor Data Bus (Bayer, YUV, YCrCb, RGB)	EPDC_SDSHR	ALT3	I
		LCD_DAT22	ALT2	
		SD3_CMD	ALT3	
CSI_DATA12	Data Sensor Signal 12, part of 16-bit Sensor Data Bus (Bayer, YUV, YCrCb, RGB)	LCD_DAT21	ALT2	I
		SD3_DAT0	ALT3	
CSI_DATA13	Data Sensor Signal 13, part of 16-bit Sensor Data Bus (Bayer, YUV, YCrCb, RGB)	LCD_DAT20	ALT2	I
		SD3_DAT1	ALT3	
CSI_DATA14	Data Sensor Signal 14, part of 16-bit Sensor Data Bus (Bayer, YUV, YCrCb, RGB)	LCD_DAT19	ALT2	I
		SD3_DAT2	ALT3	
CSI_DATA15	Data Sensor Signal 15, part of 16-bit Sensor Data Bus (Bayer, YUV, YCrCb, RGB)	LCD_DAT18	ALT2	I
		SD3_DAT3	ALT3	
CSI_HSYNC (HSYNC)	Horizontal Sync (Blank Signal)	ECSPI2_MOSI	ALT3	I
		EPDC_GDOE	ALT3	
		LCD_DAT5	ALT2	
CSI_MCLK (MCLK)	"CMOS Sensor Master Clock *Note: MCLK is provided by the CCM module directly, not from the CSI module itself"	ECSPI2_MISO	ALT3	O*
		EPDC_GDRL	ALT3	
		LCD_DAT7	ALT2	
		PWM1	ALT4	
		SD2_RST	ALT4	
CSI_PIXCLK (PIXCLK)	Pixel Clock	ECSPI2_SCLK	ALT3	I
		EPDC_GDCLK	ALT3	
		LCD_DAT6	ALT2	
CSI_VSYNC (VSYNC)	Vertical Sync (Start Of Frame)	ECSPI2_SS0	ALT3	I
		EPDC_GDSP	ALT3	
		LCD_DAT4	ALT2	

## 16.3 Clocks

The following table describes the clock sources for CSI. Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 16-2. CSI Clocks**

Clock name	Clock Root	Description
csi_hclk	ahb_clk_root	Module clock
ipg_clk	ipg_clk_root	Peripheral clock
ipg_clk_s	ipg_clk_root	Peripheral access clock
ipg_clk_s_raw	ipg_clk_root	Peripheral raw data clock

## 16.4 Principles of Operation

The information found here describes the modes of operation of the sensor interface.

The CSI is designed to support generic sensor interface timing as well as CCIR656 video interface timing. Traditional CMOS sensors typically use VSYNC (SOF), HSYNC (BLANK), and PIXCLK signals to output Bayer or YUV data. Smart CMOS sensors, that come with on-chip imaging processing, usually support video mode transfer. They use an embedded timing codec to replace the VSYNC and HSYNC signal. The timing codec is defined by the CCIR656 standard.

The CSI can support connection with the sensor as follows.

- To connect with one 8-bit sensor, the sensor data interface should connect to CSI\_DATA[9:2].
- To connect with one 10-bit sensor, the sensor data interface should connect to CSI\_DATA[9:0].
- To connect with one 16-bit sensor, the sensor data interface should connect to CSI\_DATA[15:0].
- To connect with two 8-bit sensors, the sensor data interfaces should connect to CSI\_DATA[7:0] and CSI\_DATA[15:8].

### 16.4.1 Data Transfer With The Embedded DMA Controllers

The CSI has two embedded DMA controllers, one for the receive FIFO and the other for the statistic FIFO. It supports 2D DMA transfer from the receive FIFO to the frame buffers in the external memory and linear DMA transfer from the statistic FIFO.

To transfer data from the RxFIFO to the external memory, the user should set the start address in the frame buffer where the transferred data is stored, the parameters of the frame buffers, and the parameters of the image coming from the sensor. The user can have two frame buffers in the external memory. Each one will store a frame of image coming from the sensor. The embedded DMA controller will first write the frame buffer1



and then frame buffer2. These two frame buffers will be written by turns. The start address should be aligned in word and set in the CSIDMASA-FB1 and CSIDMASA-FB2 registers. In the CSIFBUF\_PARA register, the user should set the stride of the frame buffer to show how many words to skip before starting to write the next row of the image. In the CSIMAG\_PARA register, the user should set the width and height of the image coming from the sensor. The RxFF\_LEVEL and DMA\_REQ\_EN\_RFF bits in CSICR3 registers also need to be set before the data transfer starts. When the number of the data in the Rx FIFO reaches the trigger level, a DMA request will be sent to the embedded DMA controller and the data will be read out from the Rx FIFO and written through AHB bus into the external frame buffers. The burst type of transfer can be INCR4, INCR8 and INCR16 by setting DMA\_BURST\_TYPE\_RFF bits in CSICR2 register. After all data in an image frame are transferred, the DMA\_TSF\_DONE\_FB1 or DMA\_TSF\_DONE\_FB2 bit will be set in CSISR register and the interrupt can be triggered if the corresponding enable bit is set in CSICR1 register. The DMA\_REFLASH\_RFF bit in CSICR3 can be used to activate or restart the embedded DMA controller.

The Rx FIFO has the overrun protection mechanism in case the Rx FIFO is overrun during data transfer. If the Rx FIFO is full and more data needs to be received during the data transfer, the Rx FIFO will be overwritten continuously and all 128 words of data in the Rx FIFO before overrun occurred will be discarded; the corresponding 128 words memory space in the frame buffer will keep the previous values.

To transfer data from the statistic FIFO to the external memory, the user should set the start address of the external memory where the transferred data is stored and the total transfer sizes. The start address and the transfer sizes are all aligned in word and should be set in the CSIDMASA-STATFIFO and CSIDMATS-STATFIFO registers. The STATFF\_LEVEL and DMA\_REQ\_EN\_SFF bits in CSICR3 registers should also be set before the data transfer starts. When the number of the data in the STATFIFO reaches the trigger level, a dma request will be sent to the embedded DMA controller and the data will be read out from the STATFIFO and written through AHB bus into the external memory. The burst type of transfer can be INCR4, INCR8 and INCR16 by setting DMA\_BURST\_TYPE\_SFF bits in CSICR2 register. After all expected data (defined by the total transfer sizes) are transferred, the DMA\_TSF\_DONE\_SFF bit will be set in CSISR register and an interrupt can be triggered if the SFF\_DMA\_DONE\_INTEN is enabled in CSICR1 register. The DMA\_REFLASH\_SFF bit in CSICR3 can be used to activate or re-start the embedded DMA controller.

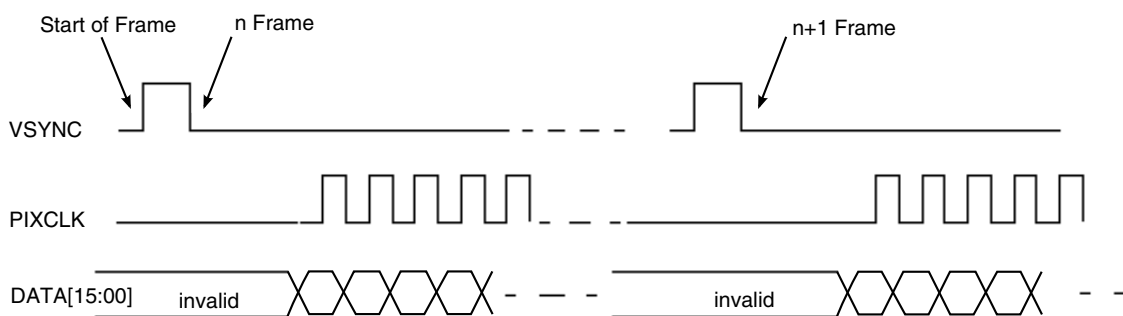
## 16.4.2 Gated Clock Mode

VSYN, HSYN, and PIXCLK signals are used in gated clock mode.

A frame starts with an active edge on VSYNC, then HSYNC asserts and holds for the entire line. The Pixel clock is valid as long as HSYNC is asserted. Data is latched at the active edge of the valid pixel clocks. HSYNC deasserts at the end of line. Pixel clocks then become invalid and CSI stops receiving data from the stream. For the next line the HSYNC timing repeats. For the next frame the VSYNC timing repeats.

### 16.4.3 Non-Gated Clock Mode

In non-gated clock mode, only the VSYNC and PIXCLK signals are used; the HSYNC signal is ignored.



**Figure 16-1. Non-Gated Clock Mode Timing Diagram**

The overall timing of non-gated mode is the same as the gated-clock mode, except for the HSYNC signal. HSYNC signal is ignored by the CSI. All incoming pixel clocks are valid and cause data to be latched into RxFIFO. The PIXCLK signal is inactive (states low) until valid data is ready to be transmitted over the bus.

Figure 16-1 shows the timing of a typical sensor. Other sensors may have the slightly different timing from that shown. The CSI can be programmed to support rising/falling-edge triggered VSYNC, active-high/low HSYNC, and rising/falling-edge triggered PIXCLK.

### 16.4.4 CCIR656 Interlace Mode

In CCIR656 mode, only the PIXCLK and CSI\_DATA[13:6] signals are used. The start of frame and blank signals are replaced by a timing codec which is embedded in the data stream. Each active line starts with an Start of Active Video (SAV) code and ends with an End of Active Video (EAV) code. In some cases, digital blanking is inserted in between EAV and SAV code. The CSI decodes and filters out the timing-coding from the data stream, recovering VSYNC and HSYNC signals for internal use, such as statistical

block control. Data is forwarded to the data receive and packing block in a sequential manner without reordering—that is, field 1 followed by field 2. The fields must be reordered in software to get back the original image.

Change of Field (COF) interrupt is triggered upon every field change. The interrupt service routine reads the status register to check for the current field.

According to the CCIR656 specification, the image must be in 625/50 PAL or 525/60 NTSC format. In addition, the image is interlaced into odd and even fields with vertical and horizontal blank data being filled into certain lines. Data must be in YCbCr422 format, each pixel contains 2 bytes, either Y + Cr or Y + Cb. These requirements are set for TV systems. The CSI module supports PAL and NTSC format only.

Figure below describes the frame structure in PAL system, showing vertical and horizontal blanking.

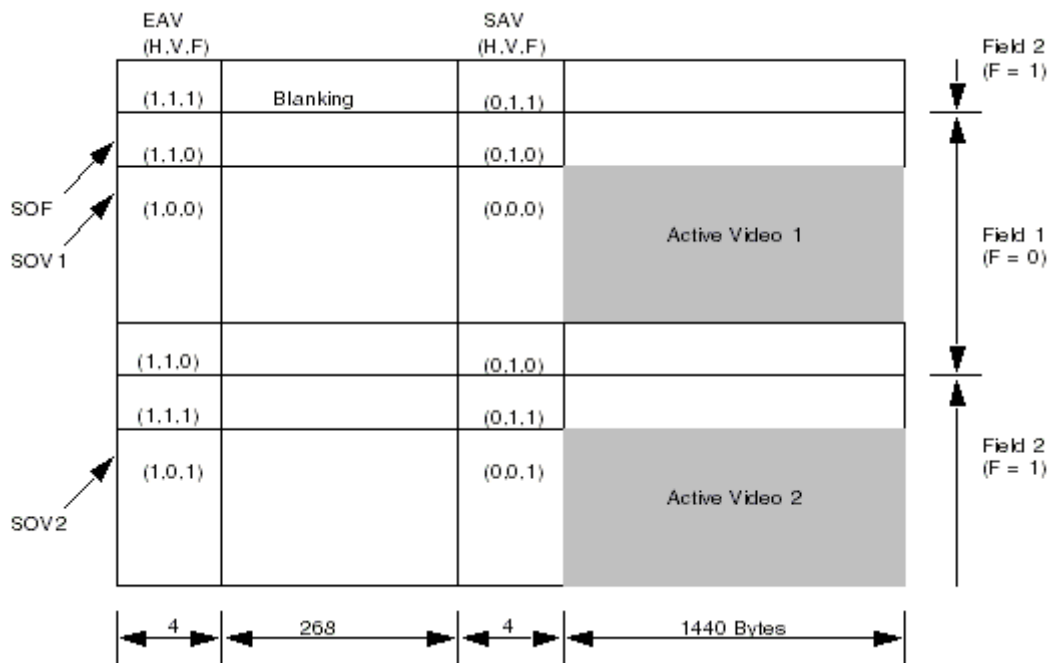


Figure 16-2. CCIR656 Interlace Mode (PAL)

Figure below describes the general timing for a single line, showing SAV and EAV.

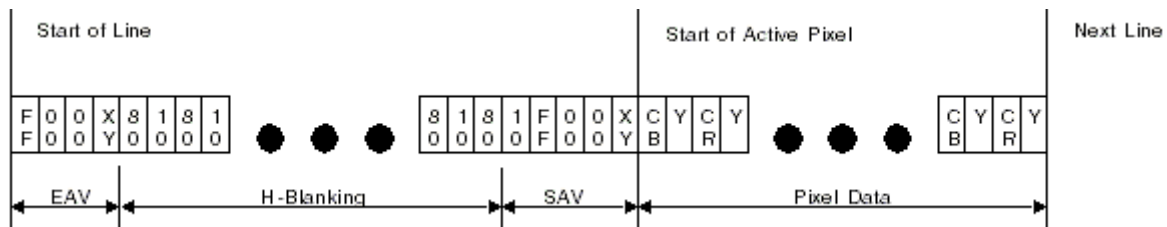


Figure 16-3. CCIR656 General Line Timing

The coding tables recommended by the CCIR656 specification are shown below. It is used in the CCIR656 mode to decode the video stream. An interrupt is generated for SOF, which is decoded from the embedded timing codec.

**Table 16-3. Coding for SAV and EAV**

Data Bit Number	1st Byte 0xFF	2nd Byte 0x00	3rd Byte 0x00	4th Byte 0xXY
7 (MSB)	1	0	0	1
6	1	0	0	F
5	1	0	0	V
4	1	0	0	H
3	1	0	0	P3
2	1	0	0	P2
1	1	0	0	P1
0	1	0	0	P0

**Table 16-4. Codes with Protection bits for Error Detection/Correction**

F	V	H	P3	P2	P1	P0
0	0	0	0	0	0	0
0	0	1	1	1	0	1
0	1	0	1	0	1	1
0	1	1	0	1	1	0
1	0	0	0	1	1	1
1	0	1	1	0	1	0
1	1	0	1	1	0	0
1	1	1	0	0	0	1

**Table 16-5. Representations by F-Bit**

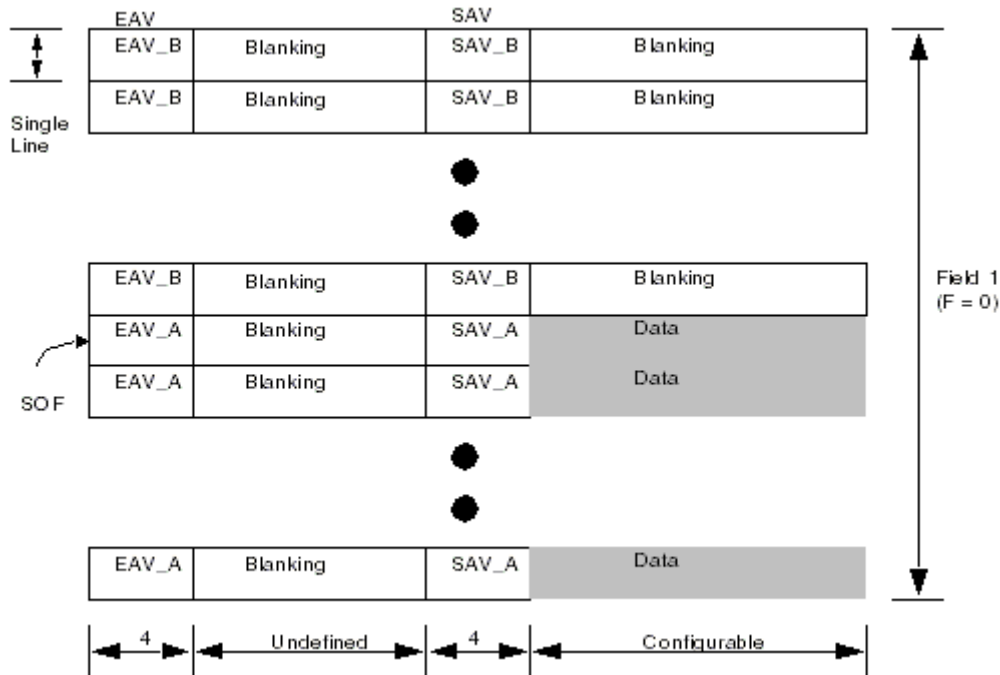
F-Bit	Representations
0	ODD FIELD (FIELD 1)
1	EVEN FIELD (FIELD 2)

### 16.4.5 CCIR656 Progressive Mode

For a CMOS camera system of VGA or CIF resolution, strict adherence to the interlace requirements stated in the CIR standard is not required.

The image is considered to have only 1 active field which is scanned in a progressive manner. This active field is regarded as field 1 and the F-bit in the timing codec is ignored by the decoder. Most sensors support CCIR timing in this mode (progressive) by default.

Figure below shows the typical flow of progressive mode.



**Figure 16-4. CCIR656 Progressive Mode (General Case)**

An interrupt is generated for SOF but not for COF. In the general case, when SOF information is retrieved from the embedded coding, it is known as internal VSYNC mode. In other cases, when the VSYNC signal is provided by the sensor, it is known as external VSYNC mode. The CSI can be operated in internal or external VSYNC mode.

### 16.4.6 Error Correction for CCIR656 Coding

According to the algorithm for CCIR coding, protection bits in the SAV and EAV are encoded in the way that allows a 1-bit error to be corrected, or a 2-bit error to be detected by the decoder. This feature is supported by the interlace mode CCIR decoder in CSI.

For the 1-bit error case, users can select the error to be corrected automatically, or simply shown as a status flag instead. For the 2-bit error case, because the decoder is unable to make a correction, the error would be shown as a status flag only.

An interrupt can be generated upon the detection of an error. This signal can be enabled or disabled without affecting the operation of the status bit.

## 16.5 Interrupt Generation

The information found here describes CSI events that generate interrupts.

### 16.5.1 Start Of Frame Interrupt (SOF\_INT)

The source of an SOF interrupt is dependent on the mode of operation.

In traditional mode, VSYNC signal is taken from sensor and SOF\_INT is generated at the rising or falling edge (programmable) of VSYNC.

In CCIR interlace mode, the SOF interrupt information is retrieved from the embedded coding and SOF\_INT is generated.

In CCIR progressive mode, there are two sources of an SOF interrupt:

- In *internal* VSYNC mode, SOF is retrieved from the embedded coding.
- In *external* VSYNC mode, VSYNC is taken from the sensor and SOF is generated at the rising edge of VSYNC.

### 16.5.2 End Of Frame Interrupt (EOF\_INT)

An EOF interrupt is generated when the frame ends and the complete frame data in RXFIFO is read.

The EOF event triggering works with the RX count register (CSIRXCNT). Software sets the RX count register to the frame size (in words). The CSI RX logic then counts the number of pixel data being received and compares it with the RX count. If the preset value is reached, an EOF interrupt is generated and the data in the RXFIFO are read. If a SOF event is detected before this happens, the EOF interrupt is not generated.

### 16.5.3 Change Of Field Interrupt (COF\_INT)

The Change of Field interrupt is only valid in CCIR Interlace mode. The COF interrupt is generated when the field toggles, either from field 1 to field 2, or field 2 to field 1.

Software should first check COF\_INT bit in the CSI Status Register (CSISTAT) before checking that F1\_INT or F2\_INT is turned on.

In PAL systems, the field changes at the beginning of the frame and coincides with SOF. For the first field, a COF interrupt is not generated, only an SOF. The COF interrupt is generated for the second field.

#### **16.5.4 CCIR Error Interrupt (ECC\_INT)**

The CCIR Error Interrupt is only valid for CCIR Interlace mode. An ECC interrupt is generated when an error is found on the SAV or EAV codes in the incoming stream. When this happens, the ECC\_INT status bit is set.

#### **16.5.5 RxFIFO Full Interrupt (RxFF\_INT)**

A RxFIFO full interrupt is generated when the number of data in RXFIFO reaches the water mark defined by RxFF\_LEVEL in CSICR3.

#### **16.5.6 Statistic FIFO Full Interrupt (STATFF\_INT)**

A StatFIFO full interrupt is generated when the number of data in STATFIFO reaches the water mark defined by STATFF\_LEVEL in CSICR3.

#### **16.5.7 RxFIFO Overrun Interrupt (RFF\_OR\_INT)**

A RxFIFO Overrun interrupt is generated when the RxFIFO has 128 words data and more data is being written in.

#### **16.5.8 Statistic FIFO Overrun Interrupt (SFF\_OR\_INT)**

A StatFIFO Overrun interrupt is generated when the STATFIFO has 64 words data and more data is being written in.

#### **16.5.9 Frame Buffer1 DMA Transfer Done Interrupt (DMA\_TSF\_DONE\_FB1)**

A DMA transfer done interrupt of frame buffer1 is generated when one frame of data are transferred from RxFIFO to the frame buffer1 in the external memory.

### **16.5.10 Frame Buffer2 DMA Transfer Done Interrupt (DMA\_TSF\_DONE\_FB2)**

A DMA transfer done interrupt of frame buffer2 is generated when one frame of data are transferred from RxFIFO to the frame buffer2 in the external memory.

### **16.5.11 Statistic FIFO DMA Transfer Done Interrupt (DMA\_TSF\_DONE\_SFF)**

A StatFIFO DMA transfer done interrupt is generated when all the data are transferred from StatFIFO to the external memory. The transfer size is defined in the STATFIFO DMA Transfer Size Register.

### **16.5.12 AHB Bus Response Error Interrupt (HRESP\_ERR\_INT)**

An AHB Bus response error interrupt is generated when a bus error is detected.

## **16.6 Data Packing Style**

Careful attention to endianness is needed given the different port sizes at different stages of the image capture path.

To enable flexible packing of image data before storage in the FIFOs, the CSI module can swap data fields by use of the PACK\_DIR and the SWAP16\_EN bit in CSI Control Register 1 (CSICR1).

The CSI module accepts 8-bit, 10-bit or 16-bit data from the sensor by configuring PIXEL\_BIT bit in CSI Control Register 1 (CSICR1) and TWO\_8BIT\_SENSOR bit in CSI Control Register3 (CSICR3). The input data is packed according to the setting of PACK\_DIR bit. The packed data is stored in the RX FIFO according to the setting of the SWAP16\_EN bit.

For 10-bit per pixel data format, each pixel is expanded to 16 bits by appending 6 zeros bits to the most significant bit. For 16-bit data format, the data path can be a combination by two 8-bit sensors. One sensor is connected to the CSI\_DATA[7:0]. The other sensor is connected to CSI\_DATA[15:8].



## 16.6.1 RX FIFO Path

### 16.6.1.1 Bayer Data

Bayer data is a type of raw data from the image sensor. This byte-wide data must be converted to the RGB space or YUV space by software. The data path for Bayer data is from the CSI to memory. If the system is in little endian, then the `PACK_DIR` bit should be set to 0. 8-bit data format from a sensor is packed to 32 bits as `P3.P2.P1.P0`, where `P0` is the pixel coming in time slot 0 (first data) and `P3` is the pixel coming in time slot 3 (the last data in the 32-bit word). When the data is addressed as bytes by software, `P0` is transferred first, `P1` is transferred next, and so on. 10-bit data format is packed to 32 bits as `000000.P1.000000.P0`, where `P0` is the 10-bit data coming in time slot 0 (first pixel) and `P1` is the 10-bit data coming in time slot 1 (second pixel). 16-bit data, from two sensors, is packed to 32 bits as `P3.P2.P1.P0`, where `P0` and `P1` are the two 8-bit data coming in time slot 0 (`P0` is the first pixel of the sensor connected with `CSI_DATA[7:0]` and `P1` is the first pixel of the sensor connected with `CSI_DATA[15:8]`). `P2` and `P3` are the two 8-bit data coming in time slot 1 (second pixels of the two sensors).

### 16.6.1.2 RGB565 Data

RGB565 data is processed data from the image sensor, which can be put directly into the display buffer. The data is 16 bits wide. The data path is from CSI to memory to the display controller. On the sensor side, data must be transmitted as `P0` first, followed by `P1`, and so on. For each pixel, whether the MSB or LSB is sent first depends on the endianness of the sensor. Data is 16 bits wide with the MSB labeled `RG`, and the LSB labeled `GB`. `P0` is represented as `RG0` and `GB0`.

CSI receives data in one of the following sequence:

- `RG0, GB0, RG1, GB1`, while `RG0` comes out at time slot 0 (first data), and `GB1` comes out at time slot 3 (last data)
- `GB0, RG0, GB1, RG1`

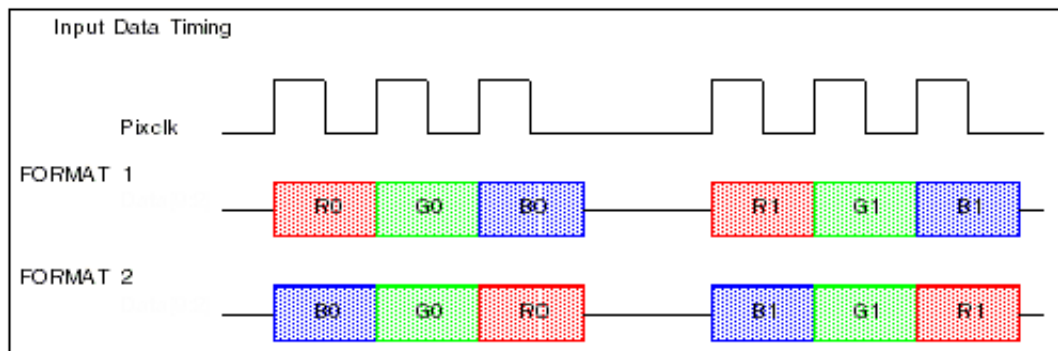
Using the first sequence as an example, and assuming the system is running in little endian, the data is presented as:

- 8-bit data from sensor: `RG0, GB0, RG1, GB1, ...`
- 32-bit data before storage in the CSI RX FIFO (`PACK_DIR` bit = 1):  
`RG0GB0RG1GB1`
- 32-bit data in CSI RX FIFO (`SWAP16_EN` bit enabled): `RG1GB1RG0GB0`

- 32-bit transfer to system memory: RG1GB1RG0GB0
- 16-bit read by display controller: RG0GB0, RG1GB1

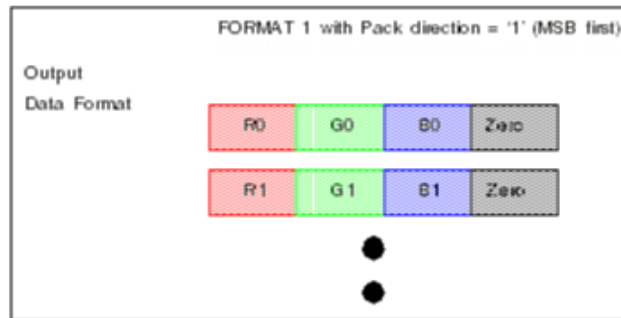
### 16.6.1.3 RGB888 Data

This is another kind of processed data from image sensor, which can be used for further image processing directly. Each of the data consist of 8-bit Red, 8-bit Green, and 8-bit Blue data. An example of timing scheme is shown in the figure below.



**Figure 16-5. Sample Timing Diagram for RGB888 Data**

An optional scheme to pack a dummy byte is provided. For every group of 3 bytes data, a dummy zero is packed to form a 32-bit word as shown in the figure below. The dummy zero is always packed at the LSB position.



## 16.6.2 STAT FIFO Path

Statistics only works for Bayer data in 8-bit per pixel format. It generates 16-bit statistical output from the 8-bit Bayer input (CSI\_DATA[13:6]). The outputs are Sum of Green (G), Sum of Red (R), Sum of Blue (B), and Auto Focus (F). Each output is 16-bits wide.

The settings of PACK\_DIR and SWAP16\_EN bits in the CSICR1 register have no effect on the input path. The PACK\_DIR only controls how the 16-bit stat output is packed into the 32-bit STAT FIFO.

When the PACK\_DIR bit = 1, the stat data is packed as:

First 32-bit: RG

Second 32-bit: BF

...

When the PACK\_DIR bit = 0, the stat data is packed as:

First 32-bit GR

Second 32-bit: FB

...

## 16.7 CSI Memory Map/Register Definition

All the 32-bit registers of the CSI module are summarized in the Memory Map below:

**CSI memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20E_4000	CSI Control Register 1 (CSI_CSICR1)	32	R/W	4000_0800h	<a href="#">16.7.1/625</a>
20E_4004	CSI Control Register 2 (CSI_CSICR2)	32	R/W	0000_0000h	<a href="#">16.7.2/629</a>
20E_4008	CSI Control Register 3 (CSI_CSICR3)	32	R/W	0000_0000h	<a href="#">16.7.3/631</a>
20E_400C	CSI Statistic FIFO Register (CSI_CSISTATFIFO)	32	R	0000_0000h	<a href="#">16.7.4/633</a>
20E_4010	CSI RX FIFO Register (CSI_CSIRFIFO)	32	R	0000_0000h	<a href="#">16.7.5/634</a>
20E_4014	CSI RX Count Register (CSI_CSIRXCNT)	32	R/W	0000_9600h	<a href="#">16.7.6/634</a>
20E_4018	CSI Status Register (CSI_CSISR)	32	R/W	0000_4000h	<a href="#">16.7.7/635</a>

*Table continues on the next page...*

## CSI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20E_4020	CSI DMA Start Address Register - for STATFIFO (CSI_CSIDMASA_STATFIFO)	32	R/W	0000_0000h	<a href="#">16.7.8/638</a>
20E_4024	CSI DMA Transfer Size Register - for STATFIFO (CSI_CSIDMATS_STATFIFO)	32	R/W	0000_0000h	<a href="#">16.7.9/638</a>
20E_4028	CSI DMA Start Address Register - for Frame Buffer1 (CSI_CSIDMASA_FB1)	32	R/W	0000_0000h	<a href="#">16.7.10/639</a>
20E_402C	CSI DMA Transfer Size Register - for Frame Buffer2 (CSI_CSIDMASA_FB2)	32	R/W	0000_0000h	<a href="#">16.7.11/640</a>
20E_4030	CSI Frame Buffer Parameter Register (CSI_CSIFBUF_PARA)	32	R/W	0000_0000h	<a href="#">16.7.12/640</a>
20E_4034	CSI Image Parameter Register (CSI_CSIIMAG_PARA)	32	R/W	0000_0000h	<a href="#">16.7.13/641</a>

## 16.7.1 CSI Control Register 1 (CSI\_CSICR1)

This register controls the sensor interface timing and interrupt generation. The interrupt enable bits in this register control the interrupt signals and the status bits. That means status bits will only function when the corresponding interrupt bits are enabled.

Address: 20E\_4000h base + 0h offset = 20E\_4000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SWAP16_EN	EXT_VSYNC	EOF_INT_EN	PtP_IF_EN	CCIR_MODE	COF_INT_EN	SF_OR_INTEN	RF_OR_INTEN	Reserved	SFF_DMA_DONE_INTEN	STATFF_INTEN	FB2_DMA_DONE_INTEN	FB1_DMA_DONE_INTEN	RXFF_INTEN	SOF_POL	SOF_INTEN
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				HSYNC_POL	CCIR_EN	Reserved	FCC	PACK_DIR	CLR_STATFIFO	CLR_RXFIFO	GCLK_MODE	INV_DATA	INV_PCLK	REDGE	PIXEL_BIT
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0

## CSI\_CSICR1 field descriptions

Field	Description
31 SWAP16_EN	<p>SWAP 16-Bit Enable. This bit enables the swapping of 16-bit data. Data is packed from 8-bit or 10-bit to 32-bit first (according to the setting of PACK_DIR) and then swapped as 16-bit words before being put into the RX FIFO. The action of the bit only affects the RX FIFO and has no affect on the STAT FIFO.</p> <p><b>NOTE:</b> Example of swapping enabled: Data input to FIFO = 0x11223344 Data in RX FIFO = 0x 33441122</p> <p><b>NOTE:</b> Example of swapping disabled: Data input to FIFO = 0x11223344 Data in RX FIFO = 0x11223344</p> <p>0   Disable swapping 1   Enable swapping</p>
30 EXT_VSYNC	<p>External VSYNC Enable. This bit controls the operational VSYNC mode.</p> <p><b>NOTE:</b> This only works when the CSI is in CCIR progressive mode.</p> <p>0   Internal VSYNC mode 1   External VSYNC mode</p>
29 EOF_INT_EN	<p>End-of-Frame Interrupt Enable. This bit enables and disables the EOF interrupt.</p> <p>0   EOF interrupt is disabled. 1   EOF interrupt is generated when RX count value is reached.</p>
28 PrP_IF_EN	<p>CSI-PrP Interface Enable. This bit controls the CSI to PrP bus. When enabled the RxFIFO is detached from the AHB bus and connected to PrP. All CPU reads or DMA accesses to the RxFIFO register are ignored. All CSI interrupts are also masked.</p> <p>0   CSI to PrP bus is disabled 1   CSI to PrP bus is enabled</p>
27 CCIR_MODE	<p>CCIR Mode Select. This bit controls the CCIR mode of operation.</p> <p>This bit only works in CCIR interface mode.</p> <p>0   Progressive mode is selected 1   Interlace mode is selected</p>
26 COF_INT_EN	<p>Change Of Image Field (COF) Interrupt Enable. This bit enables the COF interrupt.</p> <p>This bit works only in CCIR interlace mode which is when CCIR_EN = 1 and CCIR_MODE = 1.</p> <p>0   COF interrupt is disabled 1   COF interrupt is enabled</p>
25 SF_OR_INTEN	<p>STAT FIFO Overrun Interrupt Enable. This bit enables the STATFIFO overrun interrupt.</p> <p>0   STATFIFO overrun interrupt is disabled 1   STATFIFO overrun interrupt is enabled</p>
24 RF_OR_INTEN	<p>RxFIFO Overrun Interrupt Enable. This bit enables the RX FIFO overrun interrupt.</p> <p>0   RxFIFO overrun interrupt is disabled 1   RxFIFO overrun interrupt is enabled</p>

*Table continues on the next page...*

**CSI\_CSICR1 field descriptions (continued)**

Field	Description
23 -	This field is reserved. Reserved. This bit is reserved and should read 0.
22 SFF_DMA_ DONE_INTEN	STATFIFO DMA Transfer Done Interrupt Enable. This bit enables the interrupt of STATFIFO DMA transfer done.  0 STATFIFO DMA Transfer Done interrupt disable 1 STATFIFO DMA Transfer Done interrupt enable
21 STATFF_INTEN	STATFIFO Full Interrupt Enable. This bit enables the STAT FIFO interrupt.  0 STATFIFO full interrupt disable 1 STATFIFO full interrupt enable
20 FB2_DMA_ DONE_INTEN	Frame Buffer2 DMA Transfer Done Interrupt Enable. This bit enables the interrupt of Frame Buffer2 DMA transfer done.  0 Frame Buffer2 DMA Transfer Done interrupt disable 1 Frame Buffer2 DMA Transfer Done interrupt enable
19 FB1_DMA_ DONE_INTEN	Frame Buffer1 DMA Transfer Done Interrupt Enable. This bit enables the interrupt of Frame Buffer1 DMA transfer done.  0 Frame Buffer1 DMA Transfer Done interrupt disable 1 Frame Buffer1 DMA Transfer Done interrupt enable
18 RXFF_INTEN	RxFIFO Full Interrupt Enable. This bit enables the RxFIFO full interrupt.  0 RxFIFO full interrupt disable 1 RxFIFO full interrupt enable
17 SOF_POL	SOF Interrupt Polarity. This bit controls the condition that generates an SOF interrupt.  0 SOF interrupt is generated on SOF falling edge 1 SOF interrupt is generated on SOF rising edge
16 SOF_INTEN	Start Of Frame (SOF) Interrupt Enable. This bit enables the SOF interrupt.  0 SOF interrupt disable 1 SOF interrupt enable
15–12 Reserved	This field is reserved. Reserved. This field is reserved.
11 HSYNC_POL	HSYNC Polarity Select. This bit controls the polarity of HSYNC. This bit only works in gated-clock-that is, GCLK_MODE = 1 and CCIR_EN = 0.  0 HSYNC is active low 1 HSYNC is active high
10 CCIR_EN	CCIR656 Interface Enable. This bit selects the type of interface used. When the CCIR656 timing decoder is enabled, it replaces the function of timing interface logic.  0 Traditional interface is selected. Timing interface logic is used to latch data. 1 CCIR656 interface is selected.
9 Reserved	This field is reserved. This field is reserved.

*Table continues on the next page...*

**CSI\_CSICR1 field descriptions (continued)**

Field	Description
8 FCC	<p>FIFO Clear Control. This bit determines how the RXFIFO and STATFIFO are cleared. When Synchronous FIFO clear is selected the RXFIFO and STATFIFO are cleared, and STAT block is reset, on every SOF. FIFOs and STAT block restarts immediately after reset. For information on the operation when Asynchronous FIFO clear is selected, refer to the descriptions for the CLR_RXFIFO and CLR_STATFIFO bits.</p> <p>0 Asynchronous FIFO clear is selected. 1 Synchronous FIFO clear is selected.</p>
7 PACK_DIR	<p>Data Packing Direction. This bit Controls how 8-bit/10-bit image data is packed into 32-bit RX FIFO, and how 16-bit statistical data is packed into 32-bit STAT FIFO.</p> <p>0 Pack from LSB first. For image data, 0x11, 0x22, 0x33, 0x44, it will appear as 0x44332211 in RX FIFO. For stat data, 0xAAAA, 0BBBBB, it will appear as 0BBBBBAAAA in STAT FIFO. 1 Pack from MSB first. For image data, 0x11, 0x22, 0x33, 0x44, it will appear as 0x11223344 in RX FIFO. For stat data, 0xAAAA, 0BBBBB, it will appear as 0AAAAABBBB in STAT FIFO.</p>
6 CLR_STATFIFO	<p>Asynchronous STATFIFO Clear. This bit clears the STATFIFO and Reset STAT block.</p> <p>This bit works only in async FIFO clear mode-that is, FCC = 0. Otherwise this bit is ignored.</p> <p>Writing 1 will clear STATFIFO and reset STAT block immediately, STATFIFO and STAT block then wait and restart after the arrival of next SOF.</p> <p>The bit is restored to 0 automatically after finish. Normally reads 0.</p>
5 CLR_RXFIFO	<p>Asynchronous RXFIFO Clear. This bit clears the RXFIFO.</p> <p>This bit works only in async FIFO clear mode-that is, FCC = 0. Otherwise this bit is ignored.</p> <p>Writing 1 clears the RXFIFO immediately, RXFIFO restarts immediately after that.</p> <p>The bit is restored to 0 automatically after finish. Normally reads 0.</p>
4 GCLK_MODE	<p>Gated Clock Mode Enable. Controls if CSI is working in gated or non-gated mode.</p> <p>This bit works only in traditional mode-that is, CCIR_EN = 0. Otherwise this bit is ignored.</p> <p>0 Non-gated clock mode. All incoming pixel clocks are valid. HSYNC is ignored. 1 Gated clock mode. Pixel clock signal is valid only when HSYNC is active.</p>
3 INV_DATA	<p>Invert Data Input. This bit enables or disables internal inverters on the data lines.</p> <p>0 CSI_D[7:0] data lines are directly applied to internal circuitry 1 CSI_D[7:0] data lines are inverted before applied to internal circuitry</p>
2 INV_PCLK	<p>Invert Pixel Clock Input. This bit determines if the Pixel Clock (CSI_PIXCLK) is inverted before it is applied to the CSI module.</p> <p>0 CSI_PIXCLK is directly applied to internal circuitry 1 CSI_PIXCLK is inverted before applied to internal circuitry</p>
1 REDGE	<p>Valid Pixel Clock Edge Select. Selects which edge of the CSI_PIXCLK is used to latch the pixel data.</p> <p>0 Pixel data is latched at the falling edge of CSI_PIXCLK 1 Pixel data is latched at the rising edge of CSI_PIXCLK</p>
0 PIXEL_BIT	<p>Pixel Bit. This bit indicates the bayer data width for each pixel. This bit should be configured before activating or re-starting the embedded DMA controller.</p> <p>0 8-bit data for each pixel 1 10-bit data for each pixel</p>



## 16.7.2 CSI Control Register 2 (CSI\_CSICR2)

This register provides the statistic block with data about which live view resolution is being used, and the starting sensor pixel of the Bayer pattern. It also contains the horizontal and vertical count used to determine the number of pixels to skip between the 64 x 64 blocks of statistics when generating statistics on live view image that are greater than 512 x 384.

Address: 20E\_4000h base + 4h offset = 20E\_4004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DMA_BURST_TYPE_RFF		DMA_BURST_TYPE_SFF		Reserved	DRM	AFS		SCE	Reserved		BTS		LVRM		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VSC								HSC							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**CSI\_CSICR2 field descriptions**

Field	Description
31–30 DMA_BURST_TYPE_RFF	Burst Type of DMA Transfer from RxFIFO. Selects the burst type of DMA transfer from RxFIFO. X0 INCR8 01 INCR4 11 INCR16
29–28 DMA_BURST_TYPE_SFF	Burst Type of DMA Transfer from STATFIFO. Selects the burst type of DMA transfer from STATFIFO. X0 INCR8 01 INCR4 11 INCR16

*Table continues on the next page...*

**CSI\_CSICR2 field descriptions (continued)**

Field	Description
27 -	This field is reserved. Reserved. These bit is reserved and should read 0.
26 DRM	Double Resolution Mode. Controls size of statistics grid.  0 Stats grid of 8 x 6 1 Stats grid of 8 x 12
25–24 AFS	Auto Focus Spread. Selects which green pixels are used for auto-focus.  00 Abs Diff on consecutive green pixels 01 Abs Diff on every third green pixels 1x Abs Diff on every four green pixels
23 SCE	Skip Count Enable. Enables or disables the skip count feature.  0 Skip count disable 1 Skip count enable
22–21 -	This field is reserved. Reserved. These bits are reserved and should read 0.
20–19 BTS	Bayer Tile Start. Controls the Bayer pattern starting point.  00 GR 01 RG 10 BG 11 GB
18–16 LVRM	Live View Resolution Mode. Selects the grid size used for live view resolution.  0 512 x 384 1 448 x 336 2 384 x 288 3 384 x 256 4 320 x 240 5 288 x 216 6 400 x 300
15–8 VSC	Vertical Skip Count. Contains the number of rows to skip. SCE must be 1, otherwise VSC is ignored.  0-255 Number of rows to skip minus 1
7–0 HSC	Horizontal Skip Count. Contains the number of pixels to skip. SCE must be 1, otherwise HSC is ignored.  0-255 Number of pixels to skip minus 1

### 16.7.3 CSI Control Register 3 (CSI\_CSICR3)

This read/write register acts as an extension of the functionality of the CSI Control register 1, adding additional control and features.

Address: 20E\_4000h base + 8h offset = 20E\_4008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FRMCNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FRMCNT_RST	DMA_REFLASH_RFF	DMA_REFLASH_SFF	DMA_REQ_EN_RFF	DMA_REQ_EN_SFF	STATFF_LEVEL			HRESP_ERR_EN	RxFF_LEVEL			TWO_8BIT_SENSOR	ZERO_PACK_EN	ECC_INT_EN	ECC_AUTO_EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### CSI\_CSICR3 field descriptions

Field	Description
31–16 FRMCNT	Frame Counter. This is a 16-bit Frame Counter (Wraps around automatically after reaching the maximum)
15 FRMCNT_RST	Frame Count Reset. Resets the Frame Counter. (Cleared automatically after reset is done) 0 Do not reset 1 Reset frame counter immediately
14 DMA_REFLASH_RFF	Reflash DMA Controller for Rx FIFO. This bit reflash the embedded DMA controller for Rx FIFO. It should be reflash before the embedded DMA controller starts to work. (Cleared automatically after reflash is done) 0 No reflash 1 Reflash the embedded DMA controller
13 DMA_REFLASH_SFF	Reflash DMA Controller for STAT FIFO. This bit reflash the embedded DMA controller for STAT FIFO. It should be reflash before the embedded DMA controller starts to work. (Cleared automatically after reflash is done) 0 No reflash 1 Reflash the embedded DMA controller

Table continues on the next page...

## CSI\_CSICR3 field descriptions (continued)

Field	Description
12 DMA_REQ_EN_RFF	DMA Request Enable for RxFIFO. This bit enables the dma request from RxFIFO to the embedded DMA controller.  0 Disable the dma request 1 Enable the dma request
11 DMA_REQ_EN_SFF	DMA Request Enable for STATFIFO. This bit enables the dma request from STATFIFO to the embedded DMA controller.  0 Disable the dma request 1 Enable the dma request
10–8 STATFF_LEVEL	STATFIFO Full Level. When the number of data in STATFIFO reach this level, STATFIFO full interrupt is generated, or STATFIFO DMA request is sent.  000 4 Words 001 8 Words 010 12 Words 011 16 Words 100 24 Words 101 32 Words 110 48 Words 111 64 Words
7 HRESP_ERR_EN	Hresponse Error Enable. This bit enables the hresponse error interrupt.  0 Disable hresponse error interrupt 1 Enable hresponse error interrupt
6–4 RxFF_LEVEL	<b>RxFIFO Full Level.</b> When the number of data in RxFIFO reaches this level, a RxFIFO full interrupt is generated, or an RXFIFO DMA request is sent.  000 4 Words 001 8 Words 010 16 Words 011 24 Words 100 32 Words 101 48 Words 110 64 Words 111 96 Words
3 TWO_8BIT_SENSOR	Two 8-bit Sensor Mode. This bit indicates one 16-bit sensor or two 8-bit sensors are connected to the 16-bit data ports. This bit should be set if there is one 16-bit sensor or two 8-bit sensors are connected. This bit should be configured before activating or restarting the embedded DMA controller.  0 Only one sensor is connected. 1 Two 8-bit sensors are connected or one 16-bit sensor is connected.
2 ZERO_PACK_EN	Dummy Zero Packing Enable. This bit causes a dummy zero to be packed with every 3 incoming bytes, forming a 32-bit word. The dummy zero is always packed to the LSB position. This packing function is only available in 8-bit/pixel mode.  0 Zero packing disabled 1 Zero packing enabled

Table continues on the next page...

**CSI\_CSICR3 field descriptions (continued)**

Field	Description
1 ECC_INT_EN	Error Detection Interrupt Enable. This bit enables and disables the error detection interrupt. This feature only works in CCIR interlace mode.  0 No interrupt is generated when error is detected. Only the status bit ECC_INT is set. 1 Interrupt is generated when error is detected.
0 ECC_AUTO_EN	Automatic Error Correction Enable. This bit enables and disables the automatic error correction. If an error occurs and error correction is disabled only the ECC_INT status bit is set. This feature only works in CCIR interlace mode.  0 Auto Error correction is disabled. 1 Auto Error correction is enabled.

**16.7.4 CSI Statistic FIFO Register (CSI\_CSISTATFIFO)**

The StatFIFO is a read-only register containing statistic data from the sensor. Writing to this register has no effect.

Address: 20E\_4000h base + Ch offset = 20E\_400Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STAT																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**CSI\_CSISTATFIFO field descriptions**

Field	Description
31–0 STAT	Static data from sensor

## 16.7.5 CSI RX FIFO Register (CSI\_CSIRFIFO)

This read-only register contains received image data. Writing to this register has no effect.

Address: 20E\_4000h base + 10h offset = 20E\_4010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IMAGE																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**CSI\_CSIRFIFO field descriptions**

Field	Description
31–0 IMAGE	Received image data

## 16.7.6 CSI RX Count Register (CSI\_CSIRXCNT)

This register works for EOF interrupt generation. It should be set to the number of words to receive that would generate an EOF interrupt.

There is an internal counter that counts the number of words read from the RX FIFO. Whenever the RX FIFO is being read, by either the CPU or the embedded DMA controller, the counter value is updated and compared with this register. If the values match, then an EOF interrupt is triggered.

Address: 20E\_4000h base + 14h offset = 20E\_4014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										RXCNT																					
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0

**CSI\_CSIRXCNT field descriptions**

Field	Description
31–22 -	This field is reserved. Reserved. These bits are reserved and should read 0.
21–0 RXCNT	RxFIFO Count. This 22-bit counter for RXFIFO is updated each time the RXFIFO is read by CPU or DMA. This counter should be set to the expected number of words to receive that would generate an EOF interrupt.

### 16.7.7 CSI Status Register (CSI\_CSISR)

This read/write register shows sensor interface status, and which kind of interrupt is being generated. The corresponding interrupt bits must be set for the status bit to function. Status bits should function normally even if the corresponding interrupt enable bits are not enabled.

Address: 20E\_4000h base + 18h offset = 20E\_4018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved						SF_OR_INT	RF_OR_INT	Reserved	DMA_TSF_DONE_SFF	STATFF_INT	DMA_TSF_DONE_FB2	DMA_TSF_DONE_FB1	RxFF_INT	EOF_INT	SOF_INT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	F2_INT	F1_INT	COF_INT	Reserved					HRESP_ERR_INT	Reserved					ECC_INT	DRDY
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**CSI\_CSISR field descriptions**

Field	Description
31–26 -	This field is reserved. Reserved. These bits are reserved and should read 0.
25 SF_OR_INT	STATFIFO Overrun Interrupt Status. Indicates the overflow status of the STATFIFO register. (Cleared by writing 1)  0 STATFIFO has not overflowed. 1 STATFIFO has overflowed.
24 RF_OR_INT	RxFIFO Overrun Interrupt Status. Indicates the overflow status of the RxFIFO register. (Cleared by writing 1)  0 RxFIFO has not overflowed. 1 RxFIFO has overflowed.

*Table continues on the next page...*

## CSI\_CSISR field descriptions (continued)

Field	Description
23 -	This field is reserved. Reserved. This bit is reserved and should read 0.
22 DMA_TSF_ DONE_SFF	DMA Transfer Done from StatFIFO. Indicates that the dma transfer from StatFIFO is completed. It can trigger an interrupt if the corresponding enable bit is set in CSICR1. This bit can be cleared by writing 1 or reflashing the StatFIFO dma controller in CSICR3.(Cleared by writing 1)  0 DMA transfer is not completed. 1 DMA transfer is completed.
21 STATFF_INT	STATFIFO Full Interrupt Status. Indicates the number of data in the STATFIFO reaches the trigger level. (this bit is cleared automatically by reading the STATFIFO)  0 STATFIFO is not full. 1 STATFIFO is full.
20 DMA_TSF_ DONE_FB2	DMA Transfer Done in Frame Buffer2. Indicates that the DMA transfer from RxFIFO to Frame Buffer2 is completed. It can trigger an interrupt if the corresponding enable bit is set in CSICR1. This bit can be cleared by by writing 1 or reflashing the RxFIFO dma controller in CSICR3. (Cleared by writing 1)  0 DMA transfer is not completed. 1 DMA transfer is completed.
19 DMA_TSF_ DONE_FB1	DMA Transfer Done in Frame Buffer1. Indicates that the DMA transfer from RxFIFO to Frame Buffer1 is completed. It can trigger an interrupt if the corresponding enable bit is set in CSICR1. This bit can be cleared by by writing 1 or reflashing the RxFIFO dma controller in CSICR3. (Cleared by writing 1)  0 DMA transfer is not completed. 1 DMA transfer is completed.
18 RxFF_INT	RxFIFO Full Interrupt Status. Indicates the number of data in the RxFIFO reaches the trigger level. (this bit is cleared automatically by reading the RxFIFO)  0 RxFIFO is not full. 1 RxFIFO is full.
17 EOF_INT	End of Frame (EOF) Interrupt Status. Indicates when EOF is detected. (Cleared by writing 1)  0 EOF is not detected. 1 EOF is detected.
16 SOF_INT	Start of Frame Interrupt Status. Indicates when SOF is detected. (Cleared by writing 1)  0 SOF is not detected. 1 SOF is detected.
15 F2_INT	CCIR Field 2 Interrupt Status. Indicates the presence of field 2 of video in CCIR mode. (Cleared automatically when current field does not match)  <b>NOTE:</b> Only works in CCIR Interlace mode.  0 Field 2 of video is not detected 1 Field 2 of video is about to start
14 F1_INT	CCIR Field 1 Interrupt Status. Indicates the presence of field 1 of video in CCIR mode. (Cleared automatically when current field does not match)  <b>NOTE:</b> Only works in CCIR Interlace mode.

*Table continues on the next page...*



**CSI\_CSISR field descriptions (continued)**

Field	Description
	0 Field 1 of video is not detected. 1 Field 1 of video is about to start.
13 COF_INT	Change Of Field Interrupt Status. Indicates that a change of the video field has been detected. Only works in CCIR Interlace mode. Software should read this bit first and then dispatch the new field from F1_INT and F2_INT. (Cleared by writing 1) 0 Video field has no change. 1 Change of video field is detected.
12–8 -	This field is reserved. Reserved. These bits are reserved and should read 0.
7 HRESP_ERR_INT	Hresponse Error Interrupt Status. Indicates that a hresponse error has been detected. (Cleared by writing 1) 0 No hresponse error. 1 Hresponse error is detected.
6–2 -	This field is reserved. Reserved. These bits are reserved and should read 0.
1 ECC_INT	CCIR Error Interrupt. This bit indicates an error has occurred. This only works in CCIR Interlace mode. (Cleared by writing 1) 0 No error detected 1 Error is detected in CCIR coding
0 DRDY	RXFIFO Data Ready. Indicates the presence of data that is ready for transfer in the RxFIFO. (Cleared automatically by reading FIFO) 0 No data (word) is ready 1 At least 1 datum (word) is ready in RXFIFO.

## 16.7.8 CSI DMA Start Address Register - for STATFIFO (CSI\_CSIDMASA\_STATFIFO)

This register provides the start address for the embedded DMA controller of STATFIFO. The embedded DMA controller will read data from STATFIFO and write it to the external memory from the start address. This register should be configured before activating or restarting the embedded DMA controller.

Address: 20E\_4000h base + 20h offset = 20E\_4020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DMA_START_ADDR_SFF															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DMA_START_ADDR_SFF															Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### CSI\_CSIDMASA\_STATFIFO field descriptions

Field	Description
31–2 DMA_START_ADDR_SFF	DMA Start Address for STATFIFO. Indicates the start address to write data. The embedded DMA controller will read data from STATFIFO and write it from this address through AHB bus. The address should be word aligned.
1–0 -	This field is reserved. Reserved. These bits are reserved and should read 0.

## 16.7.9 CSI DMA Transfer Size Register - for STATFIFO (CSI\_CSIDMATS\_STATFIFO)

This register provides the total transfer size for the embedded DMA controller of STATFIFO. This register should be configured before activating or restarting the embedded DMA controller.

Address: 20E\_4000h base + 24h offset = 20E\_4024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DMA_TSF_SIZE_SFF																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**CSI\_CSIDMATS\_STATFIFO field descriptions**

Field	Description
31–0 DMA_TSF_ SIZE_SFF	DMA Transfer Size for STATFIFO. Indicates how many words to be transfered by the embedded DMA controller. The size should be word aligned.

**16.7.10 CSI DMA Start Address Register - for Frame Buffer1 (CSI\_CSIDMASA\_FB1)**

This register provides the start address in the frame buffer1 for the embedded DMA controller of RxFIFO. The embedded DMA controller will read data from RxFIFO and write it to the frame buffer1 from the start address. This register should be configured before activating or restarting the embedded DMA controller.

Address: 20E\_4000h base + 28h offset = 20E\_4028h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DMA_START_ADDR_FB1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DMA_START_ADDR_FB1															Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**CSI\_CSIDMASA\_FB1 field descriptions**

Field	Description
31–2 DMA_START_ ADDR_FB1	DMA Start Address in Frame Buffer1. Indicates the start address to write data. The embedded DMA controller will read data from RxFIFO and write it from this address through AHB bus. The address should be word aligned.
1–0 -	This field is reserved. Reserved. These bits are reserved and should read 0.

### 16.7.11 CSI DMA Transfer Size Register - for Frame Buffer2 (CSI\_CSIDMASA\_FB2)

This register provides the start address in the frame buffer2 for the embedded DMA controller of RxFIFO. The embedded DMA controller will read data from RxFIFO and write it to the frame buffer2 from the start address. This register should be configured before activating or restarting the embedded DMA controller.

Address: 20E\_4000h base + 2Ch offset = 20E\_402Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DMA_START_ADDR_FB2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DMA_START_ADDR_FB2														Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### CSI\_CSIDMASA\_FB2 field descriptions

Field	Description
31–2 DMA_START_ADDR_FB2	DMA Start Address in Frame Buffer2. Indicates the start address to write data. The embedded DMA controller will read data from RxFIFO and write it from this address through AHB bus. The address should be word aligned.
1–0 -	This field is reserved. Reserved. These bits are reserved and should read 0.

### 16.7.12 CSI Frame Buffer Parameter Register (CSI\_CSIFBUF\_PARA)

This register provides the stride of the frame buffer to show how many words to skip before starting to write the next row of the image. The width of the frame buffer minus the width of the image is the stride. This register should be configured before activating or restarting the embedded DMA controller.

Address: 20E\_4000h base + 30h offset = 20E\_4030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																FBUF_STRIDE															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**CSI\_CSIFBUF\_PARA field descriptions**

Field	Description
31–16 -	This field is reserved. Reserved. These bits are reserved and should read 0.
15–0 FBUF_STRIDE	Frame Buffer Parameter. Indicates the stride of the frame buffer. The width of the frame buffer(in word) minus the width of the image(in word) is the stride. The stride should be word aligned. The embedded DMA controller will skip the stride before starting to write the next row of the image.

**16.7.13 CSI Image Parameter Register (CSI\_CSIMAG\_PARA)**

This register provides the width and the height of the image from the sensor. The width and height should be aligned in pixel. The width of the image multiplied by the height is the total pixel size that will be transferred in a frame by the embedded DMA controller. This register should be configured before activating or restarting the embedded DMA controller.

Address: 20E\_4000h base + 34h offset = 20E\_4034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IMAGE_WIDTH																IMAGE_HEIGHT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**CSI\_CSIMAG\_PARA field descriptions**

Field	Description
31–16 IMAGE_WIDTH	Image Width. Indicates how many pixels in a line of the image from the sensor. If the input data from the sensor is 8-bit/pixel format, the IMAGE_WIDTH should be a multiple of 4 pixels. If the input data from the sensor is 10-bit/pixel or 16-bit/pixel format, the IMAGE_WIDTH should be a multiple of 2 pixels.
15–0 IMAGE_HEIGHT	Image Height. Indicates how many pixels in a column of the image from the sensor.



# Chapter 17

## Debug Monitor (DBGMON)

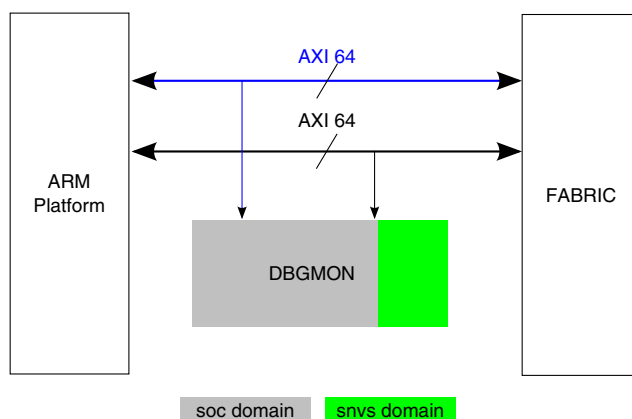
### 17.1 Overview

DBGMON is a real time AXI bus monitor and it is designed to record the last transaction on the AXI bus before the system reset. It monitors the transactions on the bus it is connected to, and records the information of them, such as address, data, ID, etc. In the i.MX 6SoloLite, DBGMON is connected to 2 AXI buses from the ARM platform.

Coordinating with WDOG, DBGMON can save the information of the selected transaction to the registers in SNVS domain, which will not lost the content when the system is powered down.

In addition, DBGMON can also be used for debugging. Various trap conditions can be set to trigger an interrupt.

This section briefly introduces the DBGMON block. The full description of the block is in [Functional Description](#). The figure below shows the DBGMON block diagram.



**Figure 17-1. DBGMON high-level diagram**

## 17.1.1 Features Summary

The key features of the DBGMON include the following:

- Save the selected transaction's information to SNVS domain registers.
- Real-time recording for most recent AXI bus transactions.
- Support out-of-order transaction, support up to 16 IDs.
- Support up to 16 outstanding transactions for each ID.
- Support various interrupts, such as address and ID trap interrupts.

## 17.1.2 Functional Description

DBGMON is designed to record the last transaction on AXI bus before the system reset.

It catch the information of the transaction and store them to SNVS domain registers which will not lost its content during reset or system power off. DBGMON uses WDOG interrupt as the flag indicating the system is going to reset.

In order to store the information when SOC power is off, DBGMON has 2 power domains - SOC power domain and SNVS power domain. In SOC power domain part of DBGMON, a memory with depth 4 will keep recording the transactions on the bus. Once the WDOG interrupt was received, one transaction's info(selected by REQ\_SEL field of HW\_DBGMON\_CTRL) will be snapped to SNVS domain registers.

Since DBGMON needs time to snap the transaction to SNVs domain registers, for WDOG, there should be an time duration between interrupt and ipp\_wdog singals(Any time duration except zero seconds is OK).

Later, when the system go out of the reset, you can read out the content of the information of the transaction from SNVS domain registers. The information includes: Address, Data, ID, RDWR(read/write attribute), COMPLETE(whether this transaction completes).

## 17.1.3 Application

Figure below shows the flowchat of the operation of DBGMON, coordiating with WDOG.



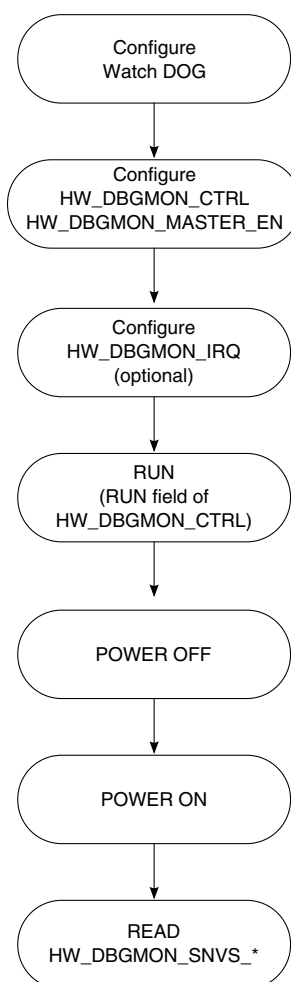


Figure 17-2. Flowchat of the DBGMON operation

## 17.2 DBGMON Memory Map/Register Definition

This section includes the block memory map and detailed descriptions of all registers. This table lists the DBGMON registers and their offsets.

### DBGMON memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
412_0000	HW_DBGMON_CTRL (DBGMON_HW_DBGMON_CTRL)	32	R/W	C000_0000h	<a href="#">17.2.1/646</a>

*Table continues on the next page...*

## DBGMON memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
412_0010	HW_DBGMON_MASTER_EN (DBGMON_HW_DBGMON_MASTER_EN)	32	R/W	0000_0000h	<a href="#">17.2.2/648</a>
412_0020	HW_DBGMON_IRQ (DBGMON_HW_DBGMON_IRQ)	32	R/W	0000_FFFFh	<a href="#">17.2.3/649</a>
412_0030	HW_DBGMON_TRAP_ADDR_LOW (DBGMON_HW_DBGMON_TRAP_ADDR_LOW)	32	R/W	0000_0000h	<a href="#">17.2.4/650</a>
412_0040	HW_DBGMON_TRAP_ADDR_HIGH (DBGMON_HW_DBGMON_TRAP_ADDR_HIGH)	32	R/W	0000_0000h	<a href="#">17.2.5/651</a>
412_0050	HW_DBGMON_TRAP_ID (DBGMON_HW_DBGMON_TRAP_ID)	32	R/W	0000_0000h	<a href="#">17.2.6/651</a>
412_0060	HW_DBGMON_SNVS_ADDR (DBGMON_HW_DBGMON_SNVS_ADDR)	32	R	0000_0000h	<a href="#">17.2.7/651</a>
412_0070	HW_DBGMON_SNVS_DATA (DBGMON_HW_DBGMON_SNVS_DATA)	32	R	0000_0000h	<a href="#">17.2.8/652</a>
412_0080	HW_DBGMON_SNVS_INFO (DBGMON_HW_DBGMON_SNVS_INFO)	32	R	0000_0000h	<a href="#">17.2.9/653</a>
412_0090	HW_DBGMON_VERSION (DBGMON_HW_DBGMON_VERSION)	32	R	0101_0000h	<a href="#">17.2.10/654</a>

## 17.2.1 HW\_DBGMON\_CTRL (DBGMON\_HW\_DBGMON\_CTRL)

Address: 209\_0000h base + 209\_0000h offset = 412\_0000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SFTRST	CLKGATE	RSVD4				WDOG_IRQ_SEL		RSVD3			ADDR_MASKEN	RSVD2			WORKMODE
W																
Reset	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD1							REQSEL	RSVD0		ID_TRAPMODE	ADDR_TRAPMODE	CLR_SNVS	CLR	SNAP	RUN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## DBGMON\_HW\_DBGMON\_CTRL field descriptions

Field	Description
31 SFTRST	Set to zero for normal operation. When this bit is set to one(default), then the entire block is held in its reset state

Table continues on the next page...

**DBGMON\_HW\_DBGMON\_CTRL field descriptions (continued)**

Field	Description
	0 Allow DBGMON to operate normally 1 Hold DBGMON in reset
30 CLKGATE	This bit must be set to zero for normal operation. When set to one it gates off the clock to the block.  0 Allow DBGMON to operate normally 1 Gating off the clock of DBGMON in order to minimize the power consumption.
29–25 RSVD4	Reserved
24 WDOG_IRQ_SEL	Select the source of WDOG IRQ.  0 Select WDOG1 as WDOG IRQ source 1 Select WDOG2 as WDOG IRQ source
23–21 RSVD3	Reserved
20 ADDR_MASKEN	This field control the address mask function,  0 Address mask is disabled, all address range will be monitored 1 Address mask is enabled, address within the range defined by ADDR_HIGH, ADDR_LOW will be monitored
19–17 RSVD2	Reserved
16 WORKMODE	This field defines whether ignore the transaction in IRQ  0 The axi transaction in interrupt status will be monitored 1 The axi transaction in interrupt status will be ignored
15–10 RSVD1	Reserved
9–8 REQSEL	This field defines which sets of AXI transaction will be snapped to SNVS domain registers. Assume (N-3), (N-2), (N-1), N represent four continuous most recent AXI transactions, N is the latest transactions  00 The information of the latest transaction will be snapped to SNVS domain registers 01 (N-1) transaction will be snapped to SNVS domain registers 10 (N-2) transaction will be snapped to SNVS domain registers 11 (N-3) transaction will be snapped to SNVS domain registers
7–6 RSVD0	Reserved
5 ID_TRAPMODE	The bit defines the ID trap function.  0 ID trap function is disabled 1 ID trap function is enabled
4 ADDR_TRAPMODE	The bit defines the address trap function.  0 Address trap function is disabled 1 Address trap function is enabled
3 CLR_SNVS	Set this bit to clear the registers in SNVS domain

*Table continues on the next page...*

**DBGMON\_HW\_DBGMON\_CTRL field descriptions (continued)**

Field	Description
2 CLR	Set this bit to clear the registers in SOC domain. This bit will be automatically set to 0 once the clear process is done
1 SNAP	Set this bit to snapshot the registers selected by REQSEL to SNVS domain registers
0 RUN	Set this bit to one to enable the DBGMON operation  0 HALT, DBGMON is in halt status 1 RUN, DBGMON is in working status

## 17.2.2 HW\_DBGMON\_MASTER\_EN (DBGMON\_HW\_DBGMON\_MASTER\_EN)

Address: 209\_0000h base + 209\_0010h offset = 412\_0010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RSVD															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MID15	MID14	MID13	MID12	MID11	MID10	MID9	MID8	MID7	MID6	MID5	MID4	MID3	MID2	MID1	MID0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DBGMON\_HW\_DBGMON\_MASTER\_EN field descriptions**

Field	Description
31–16 RSVD	Reserved
15 MID15	Set to 1 to enable monitoring on MasterID 15.
14 MID14	Set to 1 to enable monitoring on MasterID 14.
13 MID13	Set to 1 to enable monitoring on MasterID 13.
12 MID12	Set to 1 to enable monitoring on MasterID 12.
11 MID11	Set to 1 to enable monitoring on MasterID 11.

Table continues on the next page...

**DBGMON\_HW\_DBGMON\_MASTER\_EN field descriptions (continued)**

Field	Description
10 MID10	Set to 1 to enable monitoring on MasterID 10.
9 MID9	Set to 1 to enable monitoring on MasterID 9.
8 MID8	Set to 1 to enable monitoring on MasterID 8.
7 MID7	Set to 1 to enable monitoring on MasterID 7.
6 MID6	Set to 1 to enable monitoring on MasterID 6.
5 MID5	Set to 1 to enable monitoring on MasterID 5.
4 MID4	Set to 1 to enable monitoring on MasterID 4.
3 MID3	Set to 1 to enable monitoring on MasterID 3.
2 MID2	Set to 1 to enable monitoring on MasterID 2.
1 MID1	Set to 1 to enable monitoring on MasterID 1.
0 MID0	Set to 1 to enable monitoring on MasterID 0.

**17.2.3 HW\_DBGMON\_IRQ (DBGMON\_HW\_DBGMON\_IRQ)**

Address: 209\_0000h base + 209\_0020h offset = 412\_0020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IRQ_MID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD												ID_TRAP_IRQ	ADDR_TRAP_IRQ	ID_TRAP_IRQEN	ADDR_TRAP_IRQEN
W																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

**DBGMON\_HW\_DBGMON\_IRQ field descriptions**

Field	Description
31–16 IRQ_MID	This field indicate which master sends the interrupt, will not update until all interrupts are cleared. each bit represents one master. IRQ_MID[16] represents Master 0 IRQ_MID[17] represents Master 1, and so on
15–4 RSVD	Reserved.
3 ID_TRAP_IRQ	This bit indicates the ID trap interrupt is happening. Write to 1 to SCT address(offset = 0x28) to clear it.
2 ADDR_TRAP_IRQ	This bit indicates the Address trap interrupt is happening. Write to 1 to SCT address(offset = 0x28) to clear it.
1 ID_TRAP_IRQEN	ID trap interrupt control. 0 ID trap interrupt is disabled 1 ID trap interrupt is enabled
0 ADDR_TRAP_IRQEN	Address trap interrupt control. 0 Address trap interrupt is disabled 1 Address trap interrupt is enabled

## 17.2.4 HW\_DBGMON\_TRAP\_ADDR\_LOW (DBGMON\_HW\_DBGMON\_TRAP\_ADDR\_LOW)

Address: 209\_0000h base + 209\_0030h offset = 412\_0030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**DBGMON\_HW\_DBGMON\_TRAP\_ADDR\_LOW field descriptions**

Field	Description
31–0 Address	This field contains 32-bit low address for the address trap range

## 17.2.5 HW\_DBGMON\_TRAP\_ADDR\_HIGH (DBGMON\_HW\_DBGMON\_TRAP\_ADDR\_HIGH)

Address: 209\_0000h base + 209\_0040h offset = 412\_0040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Address																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### DBGMON\_HW\_DBGMON\_TRAP\_ADDR\_HIGH field descriptions

Field	Description
31–0 Address	This field contains 32-bit high address for the address trap range

## 17.2.6 HW\_DBGMON\_TRAP\_ID (DBGMON\_HW\_DBGMON\_TRAP\_ID)

Address: 209\_0000h base + 209\_0050h offset = 412\_0050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TRAP_ID_HIGH																TRAP_ID_LOW															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### DBGMON\_HW\_DBGMON\_TRAP\_ID field descriptions

Field	Description
31–16 TRAP_ID_HIGH	This field contains 16-bit high ID for ID trap range
15–0 TRAP_ID_LOW	This field contains 16-bit low ID for ID trap range

## 17.2.7 HW\_DBGMON\_SNVS\_ADDR (DBGMON\_HW\_DBGMON\_SNVS\_ADDR)

Address: 209\_0000h base + 209\_0060h offset = 412\_0060h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DBGMON\_HW\_DBGMON\_SNVS\_ADDR field descriptions**

Field	Description
31–0 ADDR	This field contains 32-bit Address in SNVS domain register

## 17.2.8 HW\_DBGMON\_SNVS\_DATA (DBGMON\_HW\_DBGMON\_SNVS\_DATA)

Address: 209\_0000h base + 209\_0070h offset = 412\_0070h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

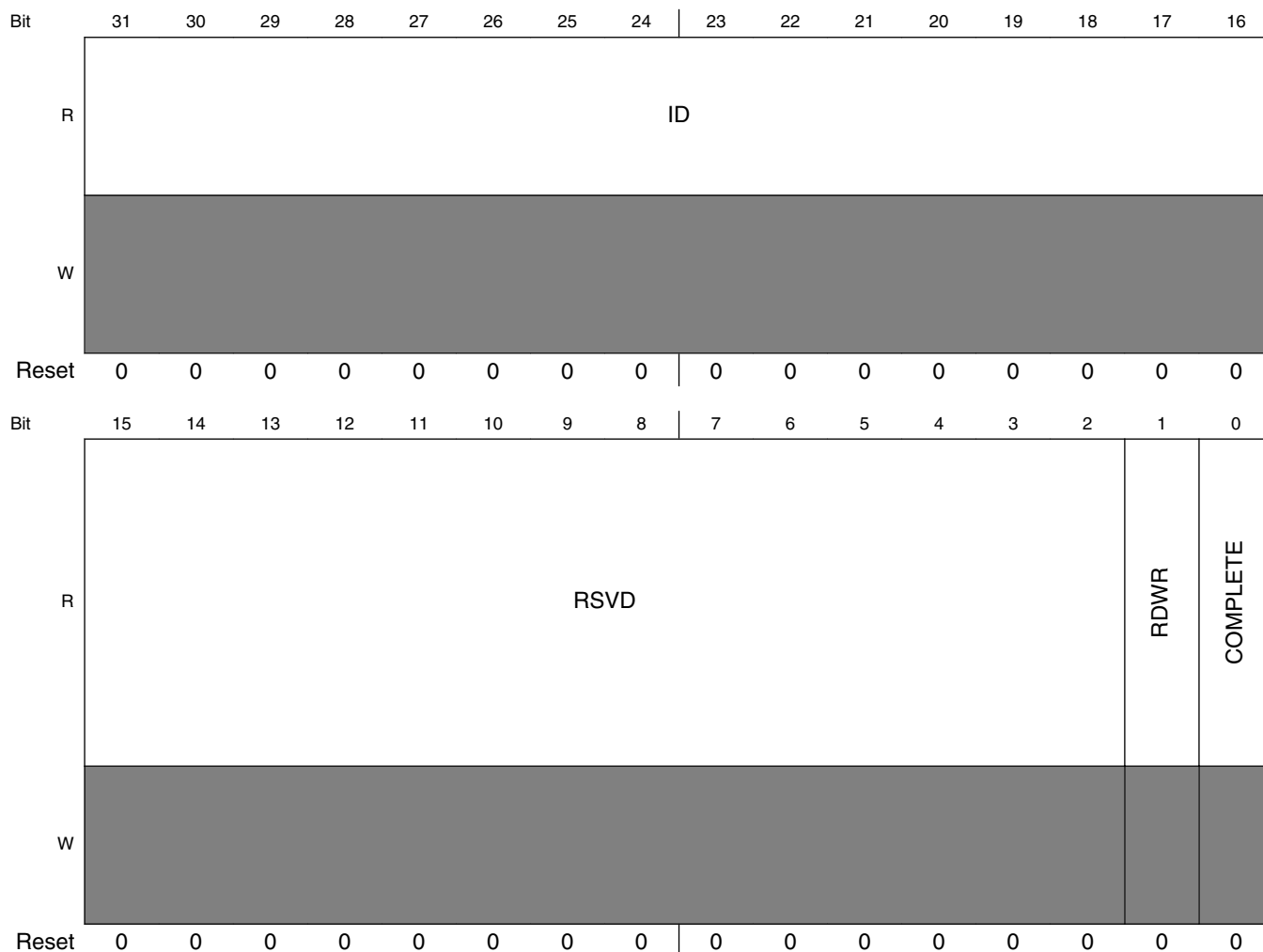
**DBGMON\_HW\_DBGMON\_SNVS\_DATA field descriptions**

Field	Description
31–0 DATA	This field contains 32-bit Data of AXI transaction in SNVS domain register



## 17.2.9 HW\_DBGMON\_SNVS\_INFO (DBGMON\_HW\_DBGMON\_SNVS\_INFO)

Address: 209\_0000h base + 209\_0080h offset = 412\_0080h



**DBGMON\_HW\_DBGMON\_SNVS\_INFO field descriptions**

Field	Description
31–16 ID	The field contain the ID of the AXI transaction in SNVS domain
15–2 RSVD	Reserved
1 RDWR	The field indicates the read/write attribute of AXI transaction in SNVS domain. 0: read, 1: Write
0 COMPLETE	The field indicates whether the AXI transaction in SNVS domain complete. 0: not complete, 1: Complete

17.2.10 HW\_DBGMON\_VERSION  
(DBGMON\_HW\_DBGMON\_VERSION)

Address: 209\_0000h base + 209\_0090h offset = 412\_0090h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJOR								MINOR								STEP															
W																																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DBGMON\_HW\_DBGMON\_VERSION field descriptions

Field	Description
31–24 MAJOR	Fixed read-only value reflecting the MAJOR field of the RTL version
23–16 MINOR	Fixed read-only value reflecting the MINOR field of the RTL version
15–0 STEP	Fixed read-only value reflecting the stepping of the RTL version

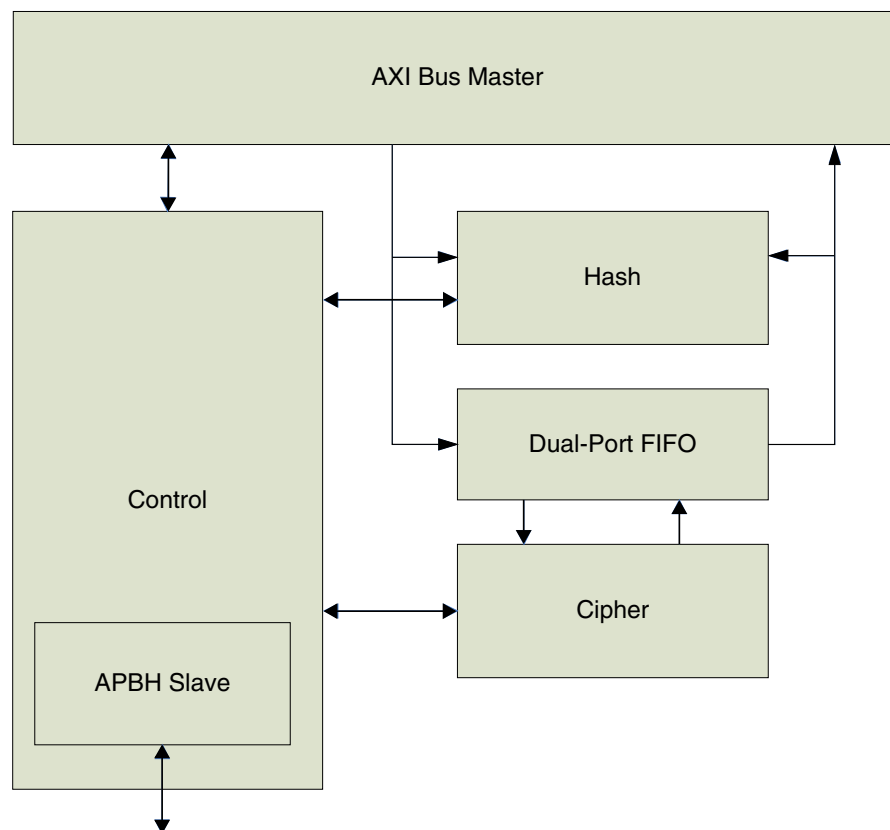
## Chapter 18

# Data Co-Processor (DCP)

### 18.1 Overview

This chapter describes the data co-processor (DCP) block included on the i.MX 6SoloLite and how to use it.

It includes sections on memory copy functionality, Advanced Encryption Standard (AES), hashing, and arbitration. Sections on programming channel operations and example code are also provided. Programmable registers are described in [DCP](#).



**Figure 18-1. Data Co-Processor (DCP) Block Diagram**

The DCP module provides support for general encryption and hashing functions typically used for security functions. Because its basic job is moving data from memory to memory, it also incorporates memory-copy (memcpy) function for both debugging and as a more efficient method of copying data between memory blocks than the DMA-based approach. The memcpy function also has a "blit" mode of operation where it can transfer a rectangular block of data to a video frame buffer.

The DCP has been designed to support a wide variety of encryption and hashing algorithms, and the control structures have been designed to allow flexibility in adding additional algorithms and modes in the future. It supports up to 16 encryption algorithms with up to 16 different modes of operation as well as up to 16 hashing algorithms. While the DCP module has been designed to support numerous algorithms, only a subset may be implemented in any given implementation (see the Capability Register).

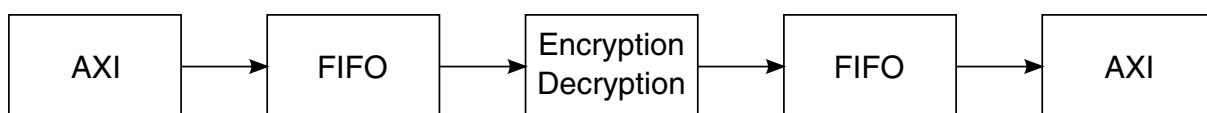
The DCP module processes data based on chained command structures written to system memory by software (in a manner similar to the DMA engine). The control block maintains registers to support four independent and concurrent chains, allowing software to virtualize access to the DCP block. Each command in a chain represents a work unit that the module will process to completion. At the end of each work unit, the control logic arbitrates among chains with outstanding commands and processes a command from that chain. Arbitration among the channels is round-robin, allowing all active channels equal access to the data engine. Each channel also supports a "high priority" mode that allows it to have priority over the remaining channels. If multiple channels are selected as high-priority, the channel arbiter selects among the high priority channels in round-robin fashion.

The data flow through the DCP module can be configured in one of five fashions, depending on the functionality activated by the control packet:

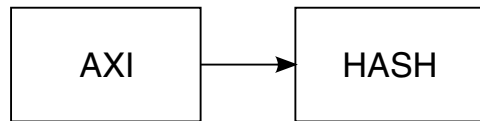
- **Memcpy/Blit Mode**-Data is moved unchanged from one memory buffer to another.



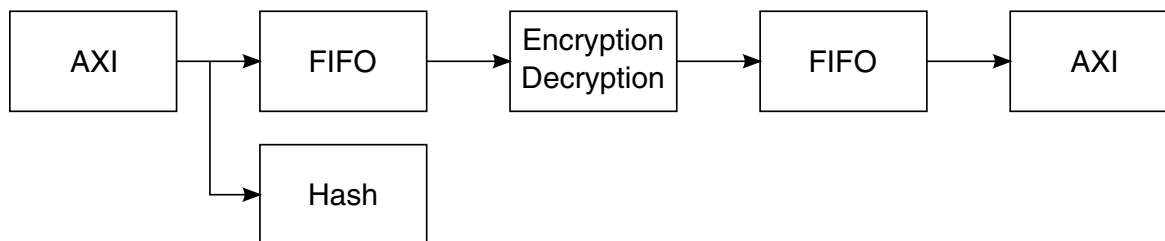
- **Encryption Only**-Data from source buffer is encrypted/decrypted into the destination buffer



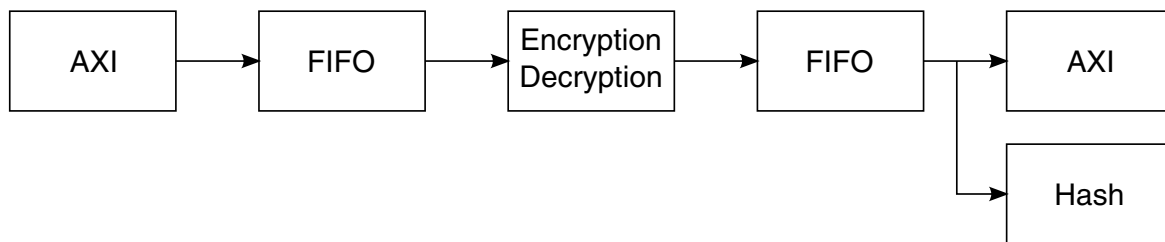
- **Hashing Only**-Data from source buffer is read, and a hash is generated.



- **Encryption and Input Hashing**-Data from source buffer is encrypted/decrypted into destination, and source buffer is hashed.



- **Encryption and Output Hashing**-Data from source buffer is encrypted/decrypted into destination, and output data is hashed.



### 18.1.1 DCP Limitations for Software

While the DCP module has been designed to be as flexible as possible, there are a few limitations to which software must adhere:

- Buffer sizes for all operations **MUST** be aligned to the natural size of the transfer algorithm used. Memcopy operations can transfer any number of bytes (one-byte granularity) and AES operations must be multiples of 16 bytes (four-word granularity). For all operations, if the byte count is not a word granularity, the hardware rounds up to the next word. Hashing is supported at a byte granularity.
- The DCP module supports buffer operations to any byte alignment, but performance will be improved if buffers are aligned to a four-byte boundary, since fetch/store operations can be performed without having to do byte operations to accommodate the misaligned addresses.
- Hash operations are limited to a 512 Mbyte buffer size. The hardware only implements a 32-bit hash length counter instead of the 64-bit counter supported by

the SHA-1/SHA-256 algorithm (counter counts bits, not bytes, therefore a total of 512 Mbytes).

- For chained hashing operations (operations involving multiple descriptors), every descriptor except the last must have a byte count that is a 16-word multiple (granularity of the hash algorithm).
- Key values cannot be written while the AES block is active. This limitation exists because the key RAM is in use while AES is operational. Any writes from the peripheral bus cannot be held in wait states; therefore, the RAM must be accessible during key writes.
- The byte-swap controls can only be used with modulo-4 length buffers. For non-modulo-4 lengths, the final partial word will contain incorrect data. Any address alignment can be used with byte swapping, however.
- The word-swap controls are only useful with cipher operations, because the logic forms the 128-bit cipher data from four words from system memory. The word-swap controls are ignored for memcpy or hashing operations.
- DCP only supports writes to word boundaries to OCRAM.

## 18.2 Clocks

The following table describes the clock sources for DCP. Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 18-1. DCP Clocks**

Clock name	Clock Root	Description
dcp_clk	ahb_clk_root	DCP clock
mem_clk	ahb_clk_root	DCP memory clock

## 18.3 Operation

The top-level DCP module contains the AXI master, peripheral slave bus interface units, the main control block and FIFO, and any encryption or hashing blocks included with the design.

The controller manages the fetching of work blocks, the fetching/storing of context information when switching between chain pointers, and the managing of the data flow through the FIFO, SHA, and AES blocks. Data entering the block from the AXI master is placed in the FIFO for consumption by the cipher block. After the cipher module has finished its operation, data is placed back into the FIFO and stored back to memory via

the AXI master. When hashing is enabled, the SHA block takes its inputs from the bus side of the FIFO to allow it to operate without waiting for the cipher block to complete. The peripheral slave provides all register controls and interfaces mainly with the control block.

### 18.3.1 Memory Copy, Blit, and Fill Functionality

In its most basic operation, the DCP supports moving unmodified data from one place in system memory to another.

This functionality is referred to as "memcpy", because it operates only on memory and it copies data from one place to another. Typical uses for memcpy might be for fast virtual memory page moves or repositioning data blocks in memory. Memcopy buffers can be aligned to any memory address and can be of any length (byte granularity). For best performance, buffers should be word-aligned, although the DCP includes enhancements to improve performance for unaligned cases.

The DCP also has the ability to do basic "blit" operations that are typical in graphics operations. To specify a blit, the control packet must have the ENABLE\_BLIT bit set in the packet control register. Blit source buffers must be contiguous. The output destination buffer for a blit operation is defined as a "M runs of N bytes" that define a rectangular region in a frame buffer. For blit operations, each line of the blit may consist of any number of bytes. After performing a "run", the DCP updates the destination pointer such that the next destination address falls on the pixel below the start of the previous run operation. This is done by incrementing the starting pointer by the frame buffer width, which is specified in the Control field.

In addition to being able to copy data within memory, the DCP also provides a "fill" operation, where source data comes not from another memory location, but from an internal register (the source buffer address in the control packet). This is done whenever the DCP\_Control0[CONSTANT\_FILL] flag is set in the packet control register (see [Control1 Field](#)). This feature may be used with memcpy to prefill memory with a specified value or during a blit operation to fill a rectangular region with a constant color.

### 18.3.2 Advanced Encryption Standard (AES)

The AES block implements a 128-bit key/data encryption/decryption block as defined by the National Institute of Standards and Technology (NIST) as US FIPS PUB 197, dated November 2001 (see references for specifications and toolkits).

### 18.3.2.1 Key Storage

The DCP implements four SRAM-based keys that may be used by software to securely store keys on a semi-permanent basis. The keys may be written via the programming interface by specifying a key index to specify which key to load and a subword pointer that indicates which word to write within the key. After a subword is written, the logic automatically increments the subword pointer so that software can program the higher-order words of the key without rewriting the key index. Keys written into the key storage are not readable.

To use a key in the key storage, the cipher descriptor packet should select the key by setting the DCP\_Control1[KEY\_SELECT] field in the Control1 descriptor field without setting the OTP\_KEY or PAYLOAD\_KEY fields in the Control0 register. See [Control0 Field](#) and [Control1 Field](#).

### 18.3.2.2 AES OTP Key

After a system reset, the OTP controller reads the e-fuse devices and provides OTP key information to the DCP. The DCP receives a 64-bit UNIQUE KEY and a 128-bit CRYPTO KEY. Depending on a key path control fuse the DCP receives the CRYPTO KEY either directly or indirectly through the SNVS trust controller module. The CRYPTO KEY in fuses is actually 256 bits and a mux is used to select the high or low 128 bits of the key. The DCP internally generates the control logic signals to capture this key into the key RAM. The system reset controller coordinates the deassertion of reset such that the OTP key is stable before the DCP reads it. If the SNVS key path is chosen then a security violation removes the CRYPTO KEY, resets the key in the DCP, and makes the key unavailable to the DCP until the next successful secure boot.

To use the OTP key, the descriptor packet should set the OTP\_KEY field in the Control1 register (see [Control1 Field](#)).

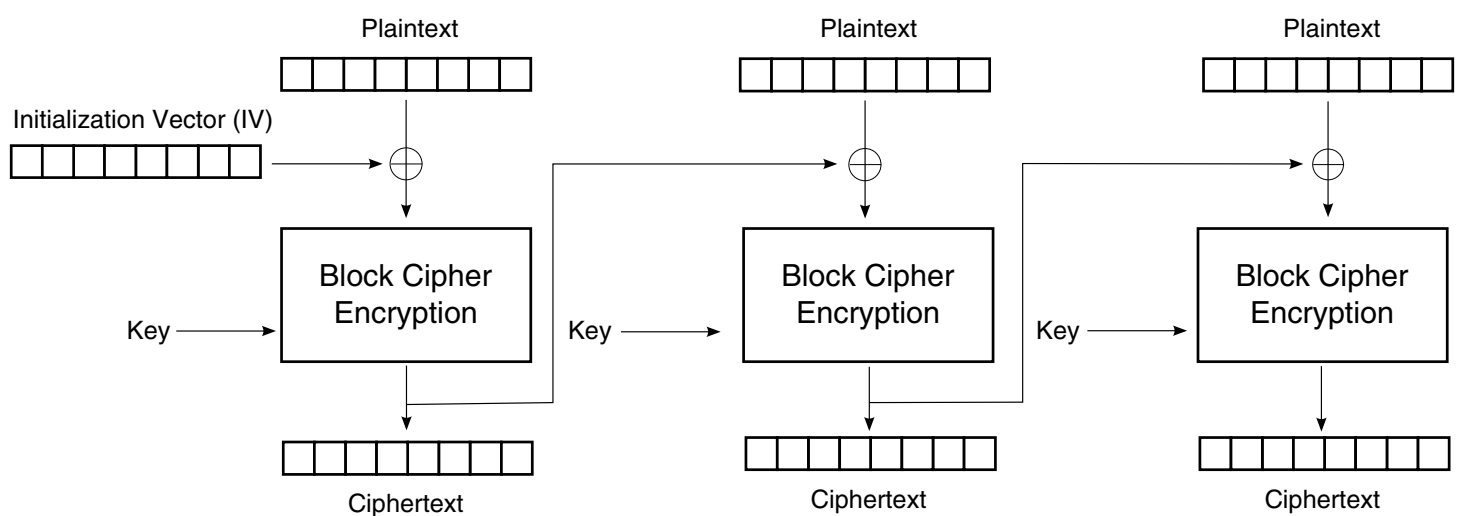
### 18.3.2.3 Encryption Modes

The most basic form of encryption is the Electronic Code Book (ECB) mode. In this mode, the encryption output is a function only of the key and the plaintext, thus it can be visualized as a giant lookup table. While this provides a great deal of security, there are a few limitations. For instance, if the same plaintext appears more than once in a block of data, the same ciphertext will also appear. This can be very evident if the plaintext contains large blocks of constant data (0s for example) and can be used to formulate attacks against a key.



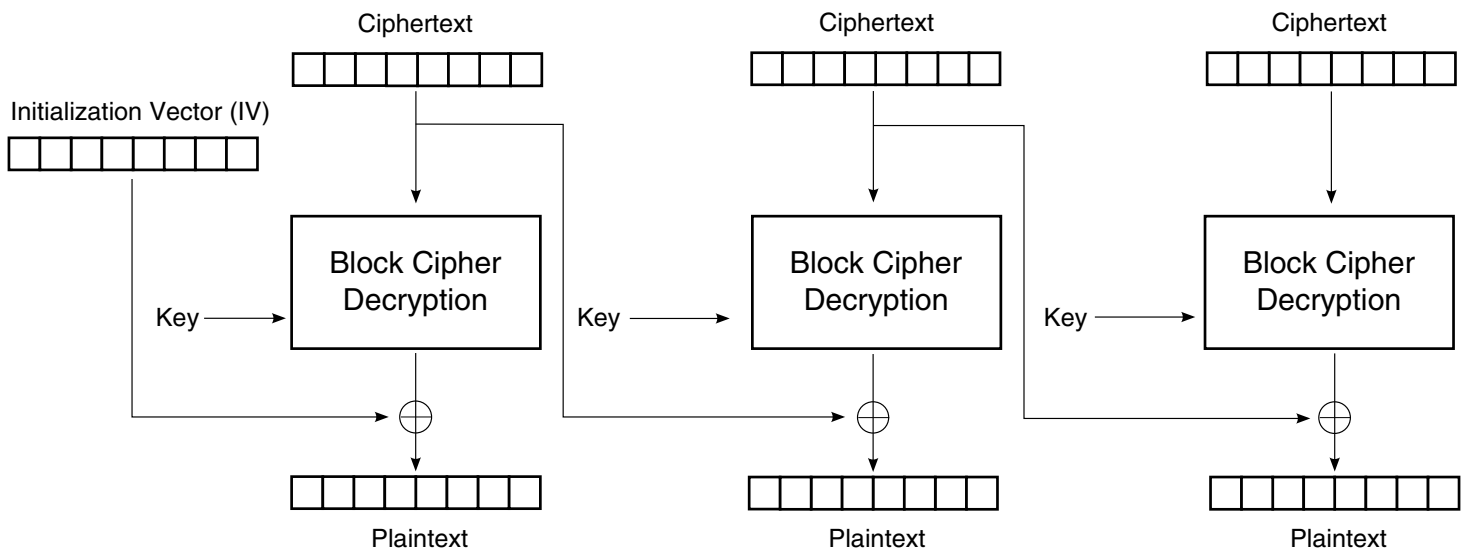
In order to make ciphers stronger, several modes of operation can be implemented around the basic ECB cipher to provide additional security. One such mode is CBC mode (Cipher Block Chaining), which takes the previous encrypted data and logically XORs it with the next incoming plaintext before performing the encryption. During decryption, the process is reversed and the previous encrypted data is XORed with the decrypted ECB data to provide the plaintext again.

The AES engine supports handling the various modes of operation. The core AES block supports ECB mode, and other algorithms are handled in the wrapper around the encryption blocks. The DCP module supports Cipher Block Chaining (CBC), which chains the data blocks by XORing the previously encrypted data with the plaintext before encryption. Cipher block chaining encryption is illustrated in [Figure 18-2](#).



**Figure 18-2. Cipher Block Chaining (CBC) Mode Encryption**

Decryption (shown in [Figure 18-3](#)) works in a similar manner, where the cipher text is first decrypted and then XORed with the previous ciphertext. For the first encryption/decryption operation, an initialization vector (IV) is used to seed the operation. The IV must be the same for both the encryption and decryption steps; otherwise, decrypted data will not match the original plaintext.



**Figure 18-3. Cipher Block Chaining (CBC) Mode Decryption**

### 18.3.3 Hashing

The hashing module implements the SHA-1 and SHA-256 hashing algorithms and a modified CRC-32 checksum algorithm.

These algorithms produce a signature for a block of data that can be used to determine whether the data is intact.

The CRC-32 algorithm implements a 32-bit CRC algorithm similar to the one used by Ethernet and many other protocols. The CRC differs from the Unix `cksum()` function in three ways:

- The CRC is initialized as 0xFFFFFFFF instead of 0x00000000.
- Logic pads zeros to a 32-bit boundary for trailing bytes.
- Logic does not post-pend the file length.

The SHA-1 block implements a 160-bit hashing algorithm that operates on 512-bit (64-byte) blocks as defined by US FIPS PUB 180-1 in 1995. The SHA-256 mode implements a 256-bit hashing algorithm that operates on 512-bit (64-byte) blocks as defined by US FIPS PUB 180-2 in 2002. The purpose of the hashing module is to generate a unique signature for a block of data that can be used to validate the integrity of the data by comparing the resulting "digest" with the original digest.

Results from hash operations are written to the beginning of the payload for the descriptor. The DCP also has the ability to check the resulting hash against a value in the payload and issue an interrupt if a mismatch occurs.

### 18.3.4 One Time Programmable (OTP) Key

After a system reset, the One Time Programmable (OTP) controller will read the e-fuse devices and provide OTP key information to the DCP through the 128-bit wide `otp_crypto_key_data` input data bus and the 64 bit wide `otp_unique_key_data` unique key input that come directly from the OTP controller.

These bus input values are held constant and originate from the eFuse shadow registers after reset. (Note that the shadow registers for these bits can be changed in the OTP block if they are unlocked.) They will only be loaded by the DCP once after the deassertion of system reset. After reset, the DCP automatically loads these wide keys 32 bits at the time into the internal key RAM.

In i.MX 6SoloLite, DCP can also get a 128-bit OCOTP key through SNVS as another option of 128-bit wide key directly from OTP controller. Setting of fuse bit "OTP\_KEY\_TO\_DCP\_DISABLE" will select the SNVS key mode, in this mode DCP will load the SNVS key with each software reset.

While OTP key is normally not available to read operations, the module will allow the key to be read if the `otp_crypto_key_ren` (read-enable) input is active (high). This allows verification of the key after it has been programmed into the eFuse device. After programming has been validated, another fuse will be set to disable the read capability.

The OTP key (crypto key) may be selected using the `DCP_Control1[OTP_KEY]` bit in the control field of the packet descriptor or by using key select `0xFF` in the `CTRL1` field of the descriptor. DCP also supports a second hardware key called the `UNIQUE_KEY` which is generated from the OTP KEY and key modifier bits from other OTP fuse fields. This key is unique to the device and may be used for encrypting private data stored on the NAND. This key may be selected by writing `0xFE` to the `KEY_SELECT` field in the `CTRL1` packet data.

### 18.3.5 Managing DCP Channel Arbitration and Performance

The DCP can have four channels all competition for DCP resources to complete their operations.

Depending on the situation, critical operations may need to be prioritized above less important operations to ensure smooth system operation. To help software achieve this goal, the DCP implements a programmable arbiter for internal DCP operations and provides "recovery" timers on each channel to throttle channel activity.

### 18.3.5.1 DCP Arbitration

The DCP implements a multi-tiered arbitration policy that allows software a lot of flexibility in scheduling DCP operations. [Figure 18-4](#) illustrates the arbitration levels and where each channel fit into the arbitration scheme.

		Channels			
		0	1	2	3
Priority Level	High	1	1	1	1
	Low	0	0	0	0

**Figure 18-4. DCP Arbitration**

Each channel can be programmed as being in either the high-priority or low-priority arbitration pool, depending on the setting in the `HIGH_PRIORITY_CHANNEL` field of the `DCP_CHANNELCTRL` register. When the corresponding bit is programmed as a 1, the channel arbitrates in the high-priority pool; otherwise it arbitrates in the low-priority pool. Once a channel has been selected, it completes one packet and then the arbiter re-arbitrates. The channel that just completed participates in the new arbitration round.

### 18.3.5.2 Channel Recovery Timers

Each channel also contains a channel recovery timer in its `DCP_CHnOPTS` register. The purpose of the recovery timer is to keep the channel inactive for a period of time after it completes an operation. This capability could be used for a high-priority channel to ensure that at least some lower-priority requests get serviced between packets or to simply allow more timeslices for other channels to perform operations. The value programmed into the recovery timers register delays the channel from operations until 16 times the programmed value. Any non-zero value should prevent the channel from participating in the next arbitration cycle.

### 18.3.6 Programming Channel Operations

The control logic block maintains the channel pointers and manages arbitration and context switching between the different channels.

It also manages the fetching of work packets and data fetch/store operations from the AXI master interface and coordinates the actions of the hashing and encryption blocks.

The control logic maintains four channels that allow software to effectively create four independent work sets for the DCP module. Software can construct chained control packets in memory that describe encryption/hashing/memcopy operations to the hardware unit. The address for this first control packet can be written to one of the four virtual channels and enabled. When one or more of the channels is enabled, the controller fetches the control packet pointed to by that channel and initiate data fetches from the source buffer. Data is then processed as described in the control packet and stored back to system memory.

Once the processing for a control packet is complete, the controller writes completion status information back to the control packet, and optionally stores relevant state information to the context buffer. If the control packet specifies a subsequent control packet, the channel's pointer is updated to the address for the next packet and an optional interrupt can be issued to the processor.

At this point, the DCP module arbitrates among the virtual channels for the next operation and processes its control packet. If a subsequent operation is continued from a previous operation, the controller automatically loads the context from the previous session into the working registers before resuming operation for that channel.

#### 18.3.6.1 Virtual Channels

The DCP module processes data via work packets stored in memory. Each of the channels contains a pointer to the current work packet and enough control logic to determine whether the channel is active and to provide status to the processor. Each channel also provides a recovery timer to help throttle processing by a particular channel. After processing a packet, the channel enables its 16-bit recovery timer (if the recovery time is set to a non-zero value). The channel will not become active again until its recovery timer has expired. The recovery timer timebase is 16 HCLK cycles, so the timer acts as a 20-bit timer with the bottom four bits implicitly tied to 0. This provides an effective range of zero to  $2^{20-1}$  clocks or 0 ns to 7.8 ms at 133 MHz.

A channel is activated any time its semaphore is non-zero and its recovery timer is cleared. The semaphore can be incremented by software to indicate that the chain pointer has been loaded with a valid pointer. As the hardware completes the work packets, it decrements the semaphore if the Decrement Semaphore flag in the Control 0 field set. Work packets may be chained together using the CHAIN or CHAIN\_CONTIGUOUS bits in the Control0 field, which causes the channel to automatically update the work packet pointer to the value in the NEXT\_COMMAND\_ADDRESS field at the end of the current work packet.

### 18.3.6.2 Context Switching

The control logic maintains four virtual channels that allow the DCP block to time-multiplex encryption, hashing, and memcpy operations, it must also retain state information when changing channels so that when a channel is resumed, it can resume the operation from where it left off. This process is called context-switching.

To minimize the number of registers used in the design, the controller saves context information from each channel into a private context area in system memory. When initializing the DCP module, software must allocate memory for the context buffer and write the address into the Context Buffer Pointer register. As the DCP module processes packets, it saves the context information for each channel to the buffer after completing each control packet. When the channel is subsequently activated, the DCP module's internal registers are then reloaded with the proper context before resuming the operation.

Each channel reserves one-fourth of the context buffer area for its context storage. The context buffer consumes 208 bytes of system memory and is formatted as shown in [Table 18-2](#).

**Table 18-2. DCP Context Buffer Layout**

Range	Channel	Data	Size
0x00-0x0C	3	Cipher Context	16 bytes
0x10-0x30	-	Hash Context	36 bytes
0x34-0x40	2	Cipher Context	16 bytes
0x44-0x64	-	Hash Context	36 bytes
0x68-0x74	1	Cipher Context	16 bytes
0x78-0x98	-	Hash Context	36 bytes
0x9C-0xA8	0	Cipher Context	16 bytes
0xAC-0xCC	-	Hash Context	36 bytes

The control logic writes to the context buffer only if the function is being used. This effectively means that the cipher context is stored for CBC encryption/decryption operations only, and the hash context is written only if SHA-1/SHA-256 is utilized. If neither of these modes are used for a given channel, the memory for the context buffer need not be allocated by software.

Since channel 0 is likely to be used for VMI in an SDRAM-based system-to-page data from the SDRAM to on-chip SRAM, the buffer allocation table has been organized so that the *highest* numbered channel uses the *lowest* area in the context buffer. For this reason, software should allocate the higher numbered channels for encryption/hashing operations and the lower numbered channels for memcpy operations to reduce the size of the context buffer.

If only a single channel is used for CBC mode operations or hashing operations, the controller provides a bit in the control register to disable context switching. In this scenario, context switching is not required, because other channels will not corrupt the state of the hashing or cipher modes. Thus, when the channel resumes, a context load is not required.

Additionally, the control logic monitors the use of CBC/hashing, so that a context reload is not done if the previous channel resumes an operation without an intermediate operation from another channel.

### 18.3.6.3 Working with Semaphores

Each channel has a semaphore register to indicate that control packets are ready to be processed. Several techniques can be used when programming the semaphores to control the execution of packets within a channel. The channel will continue to execute packets as long as the semaphore is non-zero, a chain bit is set in the descriptor, and no error has occurred.

- Software can write the number of pending packets into the semaphore register with the Decrement Semaphore bit in each control packet set. In this scenario, the channel simply decrements the semaphore for each packet set and terminates at the end of the packet chain. The benefit of this method is that software can easily determine how many packets have executed by reading the semaphore status register.
- Software can create a packet chain with the Decrement Semaphore bit set only in the last packet. In this case, software would write a 1 to the semaphore register to start the chains and the DCP will terminate after executing the last packet.
- Software can create a packet chain with the Decrement Semaphore bit set for each packet and write either a 1 or a number less than the number of packets to the semaphore register. This can be useful when debugging, because it allows the

channel to execute only a portion of the packets and software can inspect intermediate values before restarting the channel again.

If an error occurs, the channel issues an interrupt and clears the semaphore register. The channel does not perform any further operations until the error bits in the status register have been cleared. Software can manually clear a non-zero semaphore by writing 0xFF to the CLR (clear) address of the semaphore register.

#### 18.3.6.4 Work Packet Structure

Work packets for the DCP module are created in memory by the processor. Each work packet includes all information required for the hardware to process the data. The general structure of the packet is shown in [Figure 18-5](#).

Word0	Next Cmd Addr
Word1	Control0
Word2	Control1
Word3	Source Buffer Addr
Word4	Destination Buffer Addr
Word5	Buffer Size
Word6	Payload Pointer
Word7	Status

**Figure 18-5. DCP Work Packet Structure**

For some operations, additional information is required to process the data in the packet. This additional information is provided in the variable-sized payload buffer, which can be found at the address specified in the payload pointer field. When encryption is specified, the payload may include the encryption key (if the key selected resides in the packet), an initialization vector (for modes of operation such as CBC), and expected hash values when data hashing is indicated. The hardware can automatically compare the expected hash with the actual hash and interrupt the processor only on a mismatch. The payload area is used by software to store the calculated hash value at the end of hashing operations (when the HASH\_TERM control bit in DCP\_Control0 is set).



18.3.6.4.1 Next Command Address Field

The NEXT\_COMMAND\_ADDRESS field (as shown in [Table 18-3](#)) is used to point to the next work packet in the chain. This field is loaded into the channel's command pointer after the current packet has completed processing if the CHAIN bit in the CONTROL0 field (see [Table 18-4](#)) is set. The Next Command Address field should be programmed to a non-zero value when the CHAIN bit is set; otherwise, the channel will flag an invalid setup error.

Table 18-3. DCP Next Command Address Field

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
NEXT_COMMAND_ADDRESS																															

18.3.6.4.2 Control0 Field

>The main functions of the DCP module are enabled with the ENABLE\_MEMCOPY, ENABLE\_BLIT, ENABLE\_CIPHER, and ENABLE\_HASH bits from the Control0 field in the work packet.

The combinations of these bits determine the function performed by the DCP. [Table 18-5](#) summarizes the function performed for each combination.

Table 18-4. DCP Control0 Field

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

Table continues on the next page...

**Table 18-4. DCP Control0 Field (continued)**

TAG	OUTPUT_WORDSWAP
	OUTPUT_BYTESWAP
	INPUT_WORDSWAP
	INPUT_BYTESWAP
	KEY_WORDSWAP
	KEY_BYTESWA
	TEST_SEMA_IRQ
	CONSTANT_FILL
	HASH_OUTPUT
	HASH_CHECK
	HASH_TERM
	HASH_INIT
	PAYLOAD_KEY
	OTP_KEY
	CIPHER_INIT
	CIPHER_ENCRYPT
	ENABLE_BLIT
	ENABLE_HASH
	ENABLE_CIPHE
	ENABLE_MEMCOPY
	CHAIN_CONTINUOUS
	CHAIN
	DECR_SEMAPHOR
	INTERRUPT_ENABL

**Table 18-5. DCP Function Enable Bits**

Hash	Cipher	Blit	Memcopy	Description
0	0	0	X	Simple memcopy operation.
0	0	1	0	Blit operation.
0	1	0	0	Simple encrypt/decrypt operation.
1	0	0	0	Simple hash. Only reads performed.
1	0	0	1	Memcopy and hash operation.
1	1	0	0	Hash with encryption/decryption.
All Others				Invalid setup.

The CHAIN bit should be set if the NEXT\_COMMAND\_ADDRESS field has a valid pointer to the next work packet. When set, this bit causes the channel to update its pointer to the next packet. The CHAIN\_CONTINUOUS bit is similar, but it indicates that the next packet follows immediately after the current packet. This allows software to generate templates of operations without regard to the actual physical addresses used, which makes programming easier, especially in a virtual memory environment.

If the INTERRUPT\_ENABLE bit is set, the channel generates an interrupt to the processor after completing the work for this packet. When the interrupt is issued, the packet will have been completely processed, including the update of the status/payload fields in the work packet.

Each channel contains an eight-bit counting semaphore that controls whether it is in the run or idle state. When the semaphore is non-zero, the channel is ready to run and process commands. Whenever a command finishes its operation, it checks the DECR\_SEMAPHORE bit. If set, it decrements the counting semaphore. If the semaphore goes to 0 as a result, then the channel enters the idle state and remains there until the semaphore is incremented by software. When the semaphore goes to non-zero and the channel is in its idle state, it then uses the value in the NEXT\_COMMAND\_ADDRESS field to begin processing.

If the `ENABLE_CIPHER` bit is set, the data is encrypted or decrypted based on the `CIPHER_ENCRYPT` bit using the key/encryption settings from the `Control1` field. The `OTP_KEY` and `PAYLOAD_KEY` indicate the source of the keys for the operation. If the `OTP_KEY` value is set, the `KEY_SELECT` field from the `Control1` register indicates which OTP key is to be used. If the `PAYLOAD_KEY` bit is set, the first entry in the payload is the key to be used for the operation. If neither bit is set, the `KEY_SELECT` field indicates an index in the key RAM that contains the key to be used. (For cases when both the `OTP_KEY` and `PAYLOAD_KEY` bits are set, the `PAYLOAD_KEY` takes precedence.)

The `HASH_ENABLE` enables hashing of the data with the `HASH_OUTPUT` bit controlling whether the input data (`HASH_OUTPUT=0`) or output data (`HASH_OUTPUT=1`) is hashed. In addition, the hardware can be programmed to automatically check the hashed data if the `HASH_CHECK` bit is set. If the hash does not match, an interrupt is generated and the channel terminates operation on the current chain. The hashing algorithms also require two additional fields in order to operate properly. The `HASH_INIT` bit should be set for the first block in a hashing pass to properly initialize the hashing function. The `HASH_TERM` bit must be set on the final block of a hash to notify the hardware that the hashing operation is complete so that it can properly pad the tail of the buffer according to the hashing algorithm. This flag also triggers the write-back of the hash output to the work packet's payload area.

The `CONSTANT_FILL` flag may be used for both memcpy and blit operations. When this is set, the source address field is used instead as a constant that is written to all locations in the output buffer.

The `WORDSWAP` and `BYTESWAP` bits enable muxes around the FIFO to swap data to handle little-endian and big-endian storage of data in system memory. When these bits are cleared, data is assumed to be in little-endian format. When all bits are set, data is assumed to be in big-endian format.

The `TAG` field is used to identify the work packet and the associated completion status. As each packet is completed, the channel provides the status and tag information for the last packet processed.

#### 18.3.6.4.3 Control1 Field

The `Control1` field (shown in [Table 18-6](#)) provides additional values used when hashing or encrypting/decrypting data.

For blit operations, this field indicates the number of bytes in a frame buffer line, which is used to calculate the address for successive lines in each blit operation.

**Table 18-6. DCP Control1 Field**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
CIPHER_CONFIG												HASH_SELECT				KEY_SELECT								CIPHER_MODE				CIPHER_SELECT			
																FRAMEBUFFER_LENGTH (in bytes, blit mode)															

The CIPHER\_SELECT field selects from one of sixteen possible encryption algorithms (0=AES128). The CIPHER\_MODE selects the mode of operation for that cipher (0=ECB, 1=CBC).

The KEY\_SELECT field allows key selection for one of several sources. Keys can be included in the packet payload or may come from OTP or write-only registers.

The HASH\_SELECT field selects a hashing algorithm for the operation (0=SHA-1, 1=CRC32, 2=SHA-256).

The CIPHER\_CONFIG field provides optional configuration information for the selected cipher. An example would be a key length for the RC4 algorithm.

#### 18.3.6.4.4 Source Buffer

The SOURCE\_BUFFER pointer (shown in [Table 18-7](#)) specifies the location of the source buffer in memory. The buffer may reside at any byte alignment and should refer to an on-chip SRAM or off-chip SDRAM location. For optimal performance, buffers should be word aligned. When the CONSTANT\_FILL flag is set in the Control0 field, the value in this field is used as the data written to all destination buffer locations.

**Table 18-7. DCP Source Buffer Field**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
SOURCE_BUFFER																															
CONSTANT (CONSTANT_FILL mode)																															

#### 18.3.6.4.5 Destination Buffer

The DESTINATION\_BUFFER (shown in [Table 18-8](#)) specifies the location of the destination buffer in on-chip SRAM or off-chip SDRAM memory and may be set to any byte alignment. For in-place encryption, the destination buffer and source buffer should be the same value. For optimal performance, the buffer location should be word-aligned.

**Table 18-8. DCP Destination Buffer Field**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
DESTINATION_BUFFER																															

#### 18.3.6.4.6 Buffer Size Field

The BUFFER\_SIZE field (shown in [Table 18-9](#)) indicates the size of the buffers for memcpy, encryption, and hashing modes. For memcpy and hashing operations, the value may be any number of bytes (byte granularity), and for encryption modes, the length must be a multiple of the selected encryption algorithm's natural data size (16 bytes for AES).

**Table 18-9. DCP Buffer Size Field**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
NUMBER_LINES (blit mode)																BLIT_WIDTH (in bytes, blit mode)															
BUFFER_SIZE (in bytes, memcpy, encryption, hashing modes)																															

For blit operations, the buffer size field is split into two portions, a BLIT\_WIDTH value, which specifies the number of bytes in each "line" of the blit operation, and a NUMBER\_LINES value, which specifies how many vertical rows of pixels are in the image buffer.

#### 18.3.6.4.7 Payload Pointer

Some operations require additional control values that are stored in a Payload Buffer (shown in [Table 18-10](#)), which is pointed to by this field. After the DCP reads the control packet, it examines the Control0 register and determines whether any payload information is required. If so, the DCP loads the payload from the address specified in this field. (See [Payload](#) for more details.)

**Table 18-10. DCP Payload Buffer Pointer**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
PAYLOAD_POINTER																															

The payload area is also written to by the DCP at the completion of a hash operation (when the HASH\_TERM) bit is set. Software must allocate the appropriate amount of storage (20 bytes for SHA-1 and 4 bytes for CRC32) in the payload or risk having the DCP write to an unallocated address.

18.3.6.4.8 Status

After the DCP engine has completed processing of a descriptor, it writes the packet status (shown in [Table 18-11](#)) back to the descriptor In the Status field. The packet status is the value of the channel status register at the time the packet completed processing.

Table 18-11. DCP Status Field

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00
----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----	----

Table continues on the next page...

**Table 18-11. DCP Status Field (continued)**

TAG
ERROR_CODE
E
ERROR_DS
ERROR_SR
ERROR_PACKET
ERROR_SETU
HASH_MISMATCH
COMPLETE

For various error conditions, the DCP encodes additional error information in the `ERROR_CODE` field. See [DCP](#) for more details on assignments of error codes. For any completion codes besides the `COMPLETE` flag, the channel suspends processing of the command chain until the error status values are cleared in the channel status register.

The format for this field is the same as for the channel status register, with the exception that the COMPLETE bit is written to the payload status but is not present in the channel status register.

#### 18.3.6.4.9 Payload

>The payload is a variable-length field that is used to provide key, initialization values, and expected results from hashing operations.

The payload may consist of the data fields listed in [Table 18-12](#).

### Table 18-12. DCP Payload Field

Field Name	Size	Description	Condition
Cipher Key	16 bytes	AES key	Cipher enable with PAYLOAD_KEY
Cipher IV	16 bytes	CBC initialization vector	Cipher enable with CBC mode.
Hash Check	20 bytes	Hash completion value	Hash enabled with HASH_TERM fields set.

If fields are not used, they do not appear in the payload and the other payload values move upwards in the payload area. For instance, if only hashing is used, then the HASH\_CHECK value would appear at offset 0 in the payload area. [Table 18-13](#) should be used by software to determine the amount of payload to allocate and initialize.

**Table 18-13. DCP Payload Allocation by Software**

Control Bits Present			Payload Size
Hash_Check	Cipher_Init	Payload_Key	
0	0	0	0 words / 0 bytes
0	0	1	4 words / 16 bytes

Table continues on the next page...

**Table 18-13. DCP Payload Allocation by Software (continued)**

Control Bits Present			Payload Size
Hash_Check	Cipher_Init	Payload_Key	
0	1	0	4 words / 16 bytes
0	1	1	8 words / 32 bytes
1	0	0	SHA-1: 5 words / 20 bytes SHA-2: 8 words / 32 bytes
1	0	1	SHA-1: 9 words / 36 bytes SHA-2: 12 words / 48 bytes
1	1	0	SHA-1: 9 words / 36 bytes SHA-2: 12 words / 48 bytes
1	1	1	SHA-1: 13 words / 52 bytes SHA-2: 16 words / 64 bytes

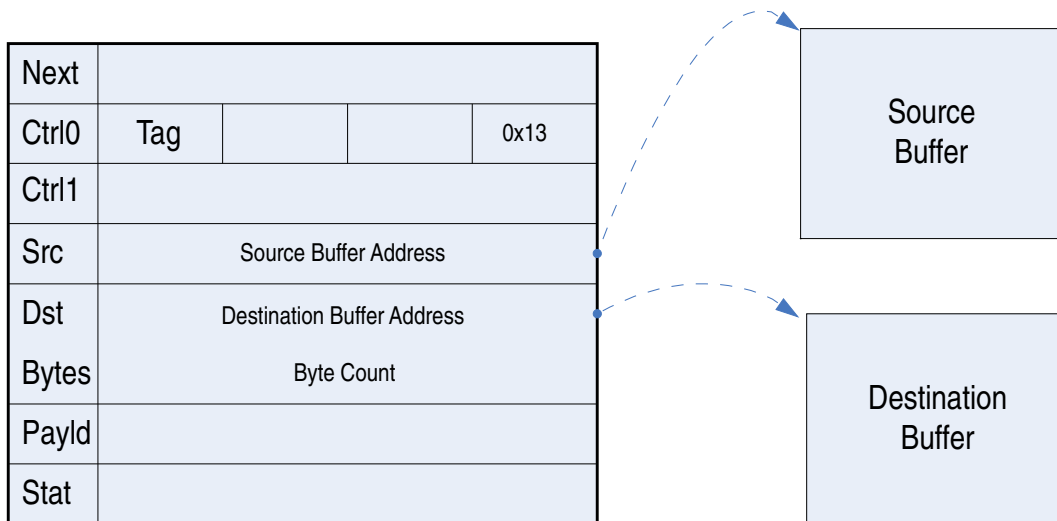
For hashing operations, the DCP module writes the final hash value back to the start of the payload area for descriptors with the HASH\_TERM bit set in the control packet. It is important that software allocate the required payload space, even though it is not required to set up the payload for control purposes.

## 18.3.7 Programming DCP Functions

### 18.3.7.1 Basic Memory Copy Programming Example

To perform a basic memcpy operation, only a single descriptor is required. The DCP simply copies data from the source to the destination buffer. This process is illustrated in [Figure 18-6](#).





0x13 = Memcopy | DecrSema | Interrupt

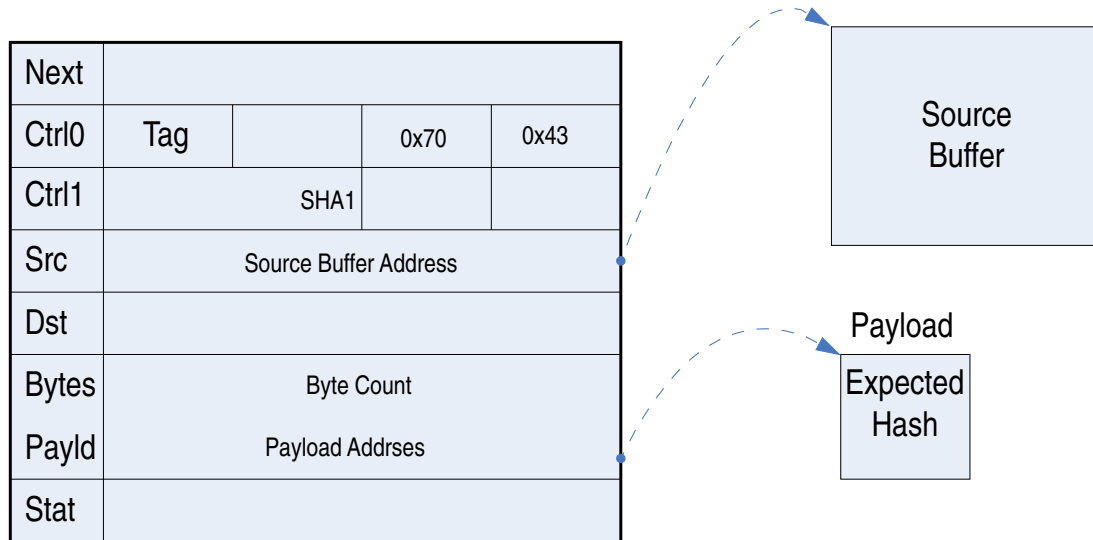
**Figure 18-6. Basic Memory Copy Operation**

```
<code>
typedef struct _dcp_descriptor
{
    u32          *next;
    hw_dcp_packet1_t  ctrl0;
    hw_dcp_packet2_t  ctrl1;
    u32          *src,
                *dst,
                buf_size,
                *payload,
                stat;
} DCP_DESCRIPTOR;

DCP_DESCRIPTOR dcp1;
u32 *srcbuffer, *dstbuffer;
// set up control packet
dcp1.next = 0; // single packet in chain
dcp1.ctrl0.U = 0; // clear ctrl0 field
dcp1.ctrl0.B.ENABLE_MEMCOPY = 1; // enable memcopy
dcp1.ctrl0.B.DECR_SEMAPHORE = 1; // decrement semaphore
dcp1.ctrl0.B.INTERRUPT = 1; // interrupt
dcp1.ctrl1.U = 0; // clear ctrl1
dcp1.src = srcbuffer; // source buffer
dcp1.dst = dstbuffer; // destination buffer
dcp1.buf_size = 512; // 512 bytes
dcp1.payload = NULL; // not required
dcp1.status = 0; // clear status
// Enable channel 0
HW_DCP_CHnCMDPTR_WR(0, dcp1); // write packet address to pointer register
HW_DCP_CHnSEMA_WR(0, 1); // increment semaphore by 1
// now wait for interrupt or poll
// polling code
while ( (HW_DCP_STAT_RD() & 0x01) == 0x00 );
// now check/clear channel status
if ( (HW_DCP_CHnSTAT_RD(0) & 0xFF) != 0 ) {
    // an error occurred
    HW_DCP_CHnSTAT_CLR(0, 0xff);
}
// clear interrupt register
HW_DCP_STAT_CLR(1);
</code>
```

### 18.3.7.2 Basic Hash Operation Programming Example

To perform a basic hash operation, only a single descriptor is required. The DCP simply reads data from the source buffer and computes the hash value on the contents. This process is illustrated in Figure 18-7.



0x70 = Hash Check | Hash Term | Hash Init

0x43 = Hash Enable | DecrSema | Interrupt

**Figure 18-7. Basic Hash Operation**

```
<code>
typedef struct _dcp_descriptor
{
    u32          *next;
    hw_dcp_packet1_t  ctrl0;
    hw_dcp_packet2_t  ctrl1;
    u32          *src,
                *dst,
                buf_size,
                *payload,
                stat;
} DCP_DESCRIPTOR;

DCP_DESCRIPTOR dcp1;
u32 *srcbuffer;
u32 payload[5];
// set up expected hash check value
payload[0]=0x01234567;
payload[1]=0x89ABCDEF;
payload[2]=0x00112233;
payload[3]=0x44556677;
payload[4]=0x8899aabb;
// set up control packet
dcp1.next = 0; // single packet in chain
dcp1.ctrl0.U = 0; // clear ctrl0 field
dcp1.ctrl0.B.HASH_CHECK = 1; // check hash when complete
dcp1.ctrl0.B.HASH_INIT = 1; // initialize hash with this block
dcp1.ctrl0.B.HASH_TERM = 1; // terminate hash with this block
dcp1.ctrl0.B.ENABLE_HASH = 1; // enable hash
dcp1.ctrl0.B.DECR_SEMAPHORE = 1; // decrement semaphore
dcp1.ctrl0.B.INTERRUPT = 1; // interrupt
dcp1.ctrl1.U = 0; // clear ctrl1
dcp1.src = srcbuffer; // source buffer
```

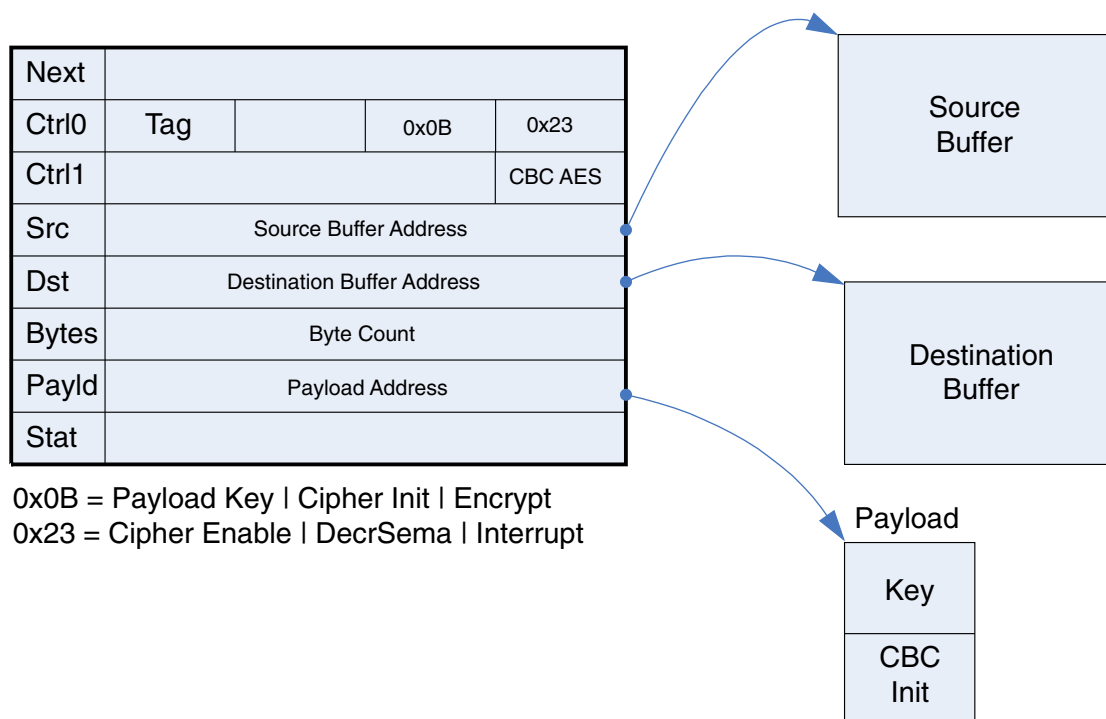
```

dcp1.dst = 0;                // no destination buffer
dcp1.buf_size = 512;         // 512 bytes
dcp1.payload = payload;      // holds expected hash value
dcp1.status = 0;             // clear status
// Enable channel 0
HW_DCP_CHnCMDPTR_WR(0, dcp1); // write packet address to pointer register
HW_DCP_CHnSEMA_WR(0, 1);      // increment semaphore by 1
// now wait for interrupt or poll
// polling code
while ( (HW_DCP_STAT_RD() & 0x01) == 0x00 );
// now check/clear channel status
if ( (HW_DCP_CHnSTAT_RD(0) & 0xFF) != 0 ) {
    // an error occurred
    HW_DCP_CHnSTAT_CLR(0, 0xff);
}
// clear interrupt register
HW_DCP_STAT_CLR(1);
</code>

```

### 18.3.7.3 Basic Cipher Operation Programming Example

To perform a basic cipher operation, only a single descriptor is required. The DCP reads data from the source buffer, encrypts it, and writes it to the destination buffer. For this example, the key is provided in the payload and the algorithm uses the AES CBC mode of operation. This requires a payload with both the key and CBC initialization value. This process is illustrated in [Figure 18-8](#).



**Figure 18-8. Basic Cipher Operation**

```

<code>
typedef struct _dcp_descriptor
{
    u32                *next;

```

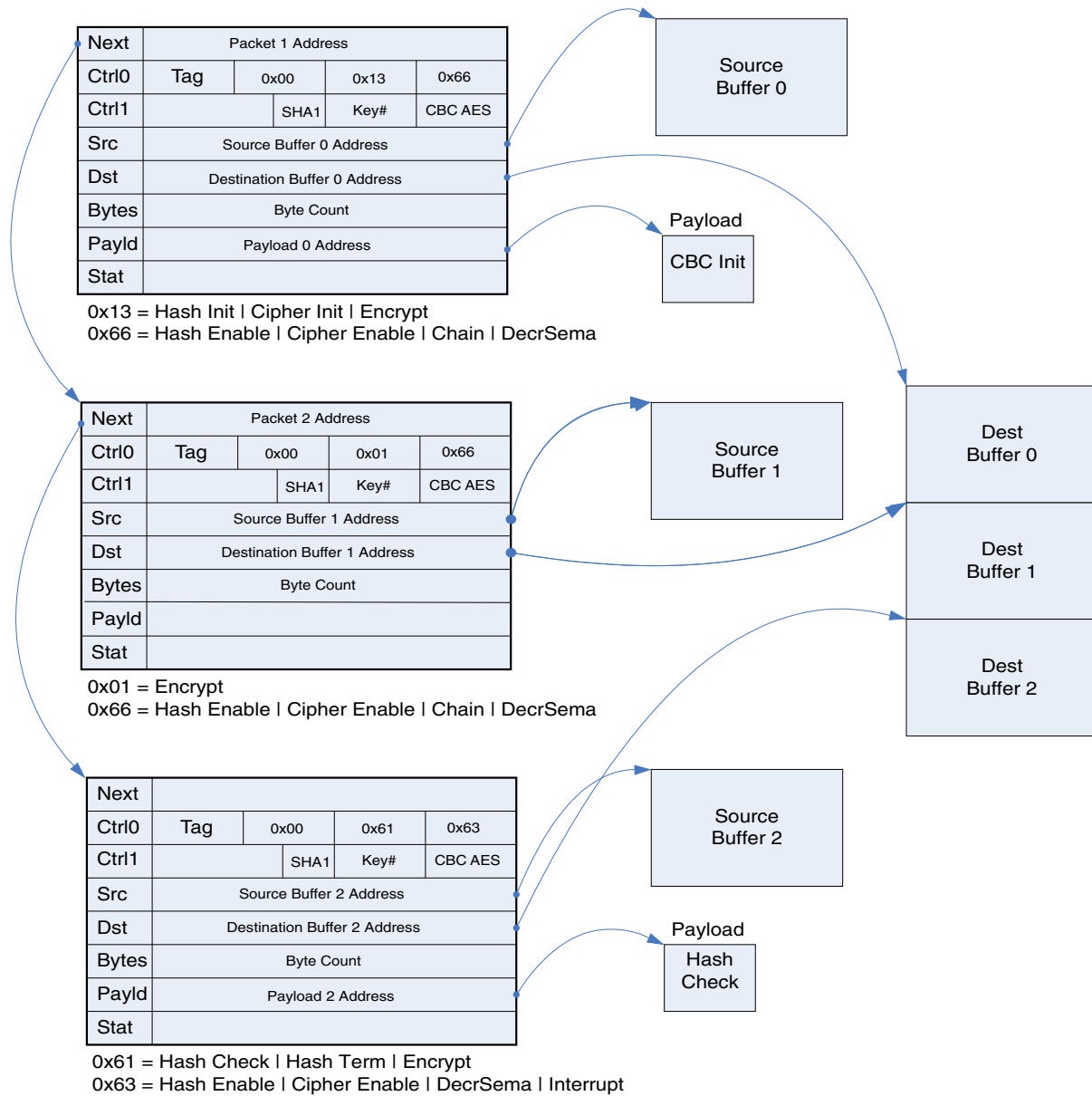
## Operation

```
hw_dcp_packet1_t  ctrl0;
hw_dcp_packet2_t  ctrl1;
u32               *src,
                  *dst,
                  buf_size,
                  *payload,
                  stat;
} DCP_DESCRIPTOR;
DCP_DESCRIPTOR dcp1;
u32 *srcbuffer;
u32 *dstbuffer;
u32 payload[8];
// set up key/CBC init in the payload
payload[0]=0x01234567;          // key
payload[1]=0x89ABCDEF;
payload[2]=0xfedcba98;
payload[3]=0x76543210;
payload[4]=0x00112233;          // CBC initialization
payload[5]=0x44556677;
payload[6]=0x8899aabb;
payload[7]=0xccddeeff;
// set up control packet
dcp1.next = 0;                  // single packet in chain
dcp1.ctrl0.U = 0;               // clear ctrl0 field
dcp1.ctrl0.B.PAYLOAD_KEY = 1;   // key is located in payload
dcp1.ctrl0.B.CIPHER_ENCRYPT = 1; // encryption operation
dcp1.ctrl0.B.CIPHER_INIT = 1;  // init CBC for this block (from payload)
dcp1.ctrl0.B.ENABLE_CIPHER = 1; // enable cipher
dcp1.ctrl0.B.DECR_SEMAPHORE = 1; // decrement semaphore
dcp1.ctrl0.B.INTERRUPT = 1;    // interrupt
dcp1.ctrl1.U = 0;              // clear ctrl1
dcp1.ctrl1.B.CIPHER_MODE = 1;  // select CBC mode of operation
dcp1.src = srcbuffer;           // source buffer
dcp1.dst = dstbuffer;           // destination buffer
dcp1.buf_size = 512;            // 512 bytes
dcp1.payload = payload;         // holds key/CBC init
dcp1.status = 0;                // clear status
// Enable channel 0
HW_DCP_CHnCMDPTR_WR(0, dcp1);  // write packet address to pointer register
HW_DCP_CHnSEMA_Wr(0, 1);        // increment semaphore by 1
// now wait for interrupt or poll
// polling code
while ( (HW_DCP_STAT_RD() & 0x01) == 0x00 );
// now check/clear channel status
if ( (HW_DCP_CHnSTAT_RD(0) & 0xFF) != 0 ) {
    // an error occurred
    HW_DCP_CHnSTAT_CLR(0, 0xff);
}
// clear interrupt register
HW_DCP_STAT_CLR(1);
</code>
```

### 18.3.7.4 Multi-Buffer Scatter/Gather Cipher and Hash Operation Programming Example

For this example, three separate buffers are encrypted and hashed with the results being directed to a unified buffer (gather operation). Three descriptors are used for the operation because there are three separate source buffer pointers. The DCP reads data from the source buffer and computes a hash on the unencrypted data. It then encrypts the data and writes it to the destination buffer. For this example, the key is located in the key RAM, and the algorithm uses the AES CBC mode of operation with a SHA-1 hash. The

payload for the first operation contains the CBC initialization value, and the payload for the last packet contains the expected hash value. The middle packet requires no payload. This process is illustrated in Figure 18-9.



**Figure 18-9. Multi-Buffer Scatter/Gather Cipher and Hash Operation**

```
<code>
typedef struct _dcp_descriptor
{
    u32          *next;
    hw_dcp_packet1_t ctrl0;
    hw_dcp_packet2_t ctrl1;
    u32          *src,
                *dst,
                buf_size,
                *payload,
                stat;
} DCP_DESCRIPTOR;
DCP_DESCRIPTOR dcp1, dcp2, dcp3;
```

## Operation

```
u32 *srcbuffer0, *srcbuffer2, *srcbuffer3;
u32 *dstbuffer;
u32 payload0[4], payload2[5];
// set up CBC init in the payload
payload0[0]=0x01234567;           // key
payload0[1]=0x89ABCDEF;
payload0[2]=0xfedcba98;
payload0[3]=0x76543210;
payload2[0]=0x00112233;           // CBC initialization
payload2[1]=0x44556677;
payload2[2]=0x8899aabb;
payload2[3]=0xccddeeff;
payload2[3]=0xaabbccdd;
// set up control packet
dcp1.next = dcp2;                 // point to packet 2
dcp1.ctrl0.U = 0;                 // clear ctrl0 field
dcp1.ctrl0.B.CIPHER_ENCRYPT = 1;   // encryption operation
dcp1.ctrl0.B.CIPHER_INIT = 1;     // init CBC for this block (from payload)
dcp1.ctrl0.B.HASH_INIT = 1;       // init hash this block
dcp1.ctrl0.B.ENABLE_CIPHER = 1;   // enable cipher
dcp1.ctrl0.B.ENABLE_HASH = 1;     // enable cipher
dcp1.ctrl0.B.DECR_SEMAPHORE = 1;  // decrement semaphore
dcp1.ctrl0.B.CHAIN = 1;           // chain to next packet
dcp1.ctrl1.U = 0;                 // clear ctrl1
dcp1.ctrl1.B.CIPHER_MODE = BV_DCP_PACKET2_CIPHER_MODE__CBC; //select CBC mode of
operation
dcp1.ctrl1.B.HASH_SELECT = BV_DCP_PACKET2_HASH_SELECT__SHA1; // select SHA1 hash
dcp1.ctrl1.B.KEY_SELECT = 2;      // select key #2
dcp1.src = srcbuffer0;            // source buffer
dcp1.dst = dstbuffer;            // destination buffer
dcp1.buf_size = 512;              // 512 bytes
dcp1.payload = payload0;          // holds key/CBC init
dcp1.status = 0;                  // clear status
// set up control packet
dcp2.next = dcp3;                 // point to packet 2
dcp2.ctrl0.U = 0;                 // clear ctrl0 field
dcp2.ctrl0.B.CIPHER_ENCRYPT = 1;   // encryption operation
dcp2.ctrl0.B.ENABLE_CIPHER = 1;   // enable cipher
dcp2.ctrl0.B.ENABLE_HASH = 1;     // enable cipher
dcp2.ctrl0.B.DECR_SEMAPHORE = 1;  // decrement semaphore
dcp2.ctrl0.B.CHAIN = 1;           // chain to next packet
dcp2.ctrl1.U = 0;                 // clear ctrl1
dcp2.ctrl1.B.CIPHER_MODE = BV_DCP_PACKET2_CIPHER_MODE__CBC; //select CBC mode of
operation
dcp2.ctrl1.B.HASH_SELECT = BV_DCP_PACKET2_HASH_SELECT__SHA1; // select SHA1 hash
dcp2.ctrl1.B.KEY_SELECT = 2;      // select key #2
dcp2.src = srcbuffer1;            // source buffer
dcp2.dst = dstbuffer+512;         // destination buffer
dcp2.buf_size = 512;              // 512 bytes
dcp2.payload = NULL;              // no payload required
dcp2.status = 0;                  // clear status
// set up control packet
dcp3.next = dcp3;                 // point to packet 2
dcp3.ctrl0.U = 0;                 // clear ctrl0 field
dcp3.ctrl0.B.HASH_TERM = 1;       // terminate hash block
dcp3.ctrl0.B.HASH_CHECK = 1;      // check hash against payload value
dcp3.ctrl0.B.CIPHER_ENCRYPT = 1;   // encryption operation
dcp3.ctrl0.B.ENABLE_CIPHER = 1;   // enable cipher
dcp3.ctrl0.B.ENABLE_HASH = 1;     // enable cipher
dcp3.ctrl0.B.DECR_SEMAPHORE = 1;  // decrement semaphore
dcp3.ctrl0.B.INTERRUPT = 1;       // interrupt on completion
dcp3.ctrl1.U = 0;                 // clear ctrl1
dcp3.ctrl1.B.CIPHER_MODE = BV_DCP_PACKET2_CIPHER_MODE__CBC; //select CBC mode of
operation
dcp3.ctrl1.B.HASH_SELECT = BV_DCP_PACKET2_HASH_SELECT__SHA1; // select SHA1 hash
dcp3.ctrl1.B.KEY_SELECT = 2;      // select key #2
dcp3.src = srcbuffer1;            // source buffer
dcp3.dst = dstbuffer+1024;        // destination buffer
dcp3.buf_size = 512;              // 512 bytes
dcp3.payload = payload2;          // payload is hash check value
```

```

dcp3.status = 0;                // clear status
// Enable channel 0
HW_DCP_CHnCMDPTR_WR(0, dcp1);  // write packet address to pointer register
HW_DCP_CHnSEMA_WR(0, 3);       // increment semaphore by 3 (for 3 packets)
// now wait for interrupt or poll
// polling code
while ( (HW_DCP_STAT_RD() & 0x01) == 0x00 );
// now check/clear channel status
if ( (HW_DCP_CHnSTAT_RD(0) & 0xFF) != 0 ) {
    // an error occurred
    HW_DCP_CHnSTAT_CLR(0, 0xff);
}
// clear interrupt register
HW_DCP_STAT_CLR(1);
</code>

```

## 18.4 DCP Memory Map/Register Definition

### DCP Hardware Register Format Summary

DCP memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20F_C000	DCP Control Register 0 (DCP_CTRL)	32	R/W	F080_0000h	<a href="#">18.4.1/684</a>
20F_C010	DCP Status Register (DCP_STAT)	32	R/W	1000_0000h	<a href="#">18.4.2/687</a>
20F_C020	DCP Channel Control Register (DCP_CHANNELCTRL)	32	R/W	0000_0000h	<a href="#">18.4.3/689</a>
20F_C030	DCP Capability 0 Register (DCP_CAPABILITY0)	32	R/W	0000_0404h	<a href="#">18.4.4/691</a>
20F_C040	DCP Capability 1 Register (DCP_CAPABILITY1)	32	R	0007_0001h	<a href="#">18.4.5/692</a>
20F_C050	DCP Context Buffer Pointer (DCP_CONTEXT)	32	R/W	0000_0000h	<a href="#">18.4.6/693</a>
20F_C060	DCP Key Index (DCP_KEY)	32	R/W	0000_0000h	<a href="#">18.4.7/693</a>
20F_C070	DCP Key Data (DCP_KEYDATA)	32	R/W	0000_0000h	<a href="#">18.4.8/694</a>
20F_C080	DCP Work Packet 0 Status Register (DCP_PACKET0)	32	R	0000_0000h	<a href="#">18.4.9/695</a>
20F_C090	DCP Work Packet 1 Status Register (DCP_PACKET1)	32	R	0000_0000h	<a href="#">18.4.10/695</a>
20F_C0A0	DCP Work Packet 2 Status Register (DCP_PACKET2)	32	R	0000_0000h	<a href="#">18.4.11/699</a>
20F_C0B0	DCP Work Packet 3 Status Register (DCP_PACKET3)	32	R	0000_0000h	<a href="#">18.4.12/700</a>
20F_C0C0	DCP Work Packet 4 Status Register (DCP_PACKET4)	32	R	0000_0000h	<a href="#">18.4.13/700</a>
20F_C0D0	DCP Work Packet 5 Status Register (DCP_PACKET5)	32	R	0000_0000h	<a href="#">18.4.14/701</a>
20F_C0E0	DCP Work Packet 6 Status Register (DCP_PACKET6)	32	R	0000_0000h	<a href="#">18.4.15/701</a>
20F_C100	DCP Channel 0 Command Pointer Address Register (DCP_CH0CMDPTR)	32	R/W	0000_0000h	<a href="#">18.4.16/702</a>

Table continues on the next page...

## DCP memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20F_C110	DCP Channel 0 Semaphore Register (DCP_CH0SEMA)	32	R/W	0000_0000h	<a href="#">18.4.17/703</a>
20F_C120	DCP Channel 0 Status Register (DCP_CH0STAT)	32	R/W	0000_0000h	<a href="#">18.4.18/704</a>
20F_C130	DCP Channel 0 Options Register (DCP_CH0OPTS)	32	R/W	0000_0000h	<a href="#">18.4.19/706</a>
20F_C140	DCP Channel 1 Command Pointer Address Register (DCP_CH1CMDPTR)	32	R/W	0000_0000h	<a href="#">18.4.20/707</a>
20F_C150	DCP Channel 1 Semaphore Register (DCP_CH1SEMA)	32	R/W	0000_0000h	<a href="#">18.4.21/708</a>
20F_C160	DCP Channel 1 Status Register (DCP_CH1STAT)	32	R/W	0000_0000h	<a href="#">18.4.22/709</a>
20F_C170	DCP Channel 1 Options Register (DCP_CH1OPTS)	32	R/W	0000_0000h	<a href="#">18.4.23/711</a>
20F_C180	DCP Channel 2 Command Pointer Address Register (DCP_CH2CMDPTR)	32	R/W	0000_0000h	<a href="#">18.4.24/712</a>
20F_C190	DCP Channel 2 Semaphore Register (DCP_CH2SEMA)	32	R/W	0000_0000h	<a href="#">18.4.25/713</a>
20F_C1A0	DCP Channel 2 Status Register (DCP_CH2STAT)	32	R/W	0000_0000h	<a href="#">18.4.26/714</a>
20F_C1B0	DCP Channel 2 Options Register (DCP_CH2OPTS)	32	R/W	0000_0000h	<a href="#">18.4.27/716</a>
20F_C1C0	DCP Channel 3 Command Pointer Address Register (DCP_CH3CMDPTR)	32	R/W	0000_0000h	<a href="#">18.4.28/717</a>
20F_C1D0	DCP Channel 3 Semaphore Register (DCP_CH3SEMA)	32	R/W	0000_0000h	<a href="#">18.4.29/718</a>
20F_C1E0	DCP Channel 3 Status Register (DCP_CH3STAT)	32	R/W	0000_0000h	<a href="#">18.4.30/719</a>
20F_C1F0	DCP Channel 3 Options Register (DCP_CH3OPTS)	32	R/W	0000_0000h	<a href="#">18.4.31/721</a>
20F_C400	DCP Debug Select Register (DCP_DBGSELECT)	32	R/W	0000_0000h	<a href="#">18.4.32/721</a>
20F_C410	DCP Debug Data Register (DCP_DBGDATA)	32	R	0000_0000h	<a href="#">18.4.33/722</a>
20F_C420	DCP Page Table Register (DCP_PAGETABLE)	32	R/W	0000_0000h	<a href="#">18.4.34/722</a>
20F_C430	DCP Version Register (DCP_VERSION)	32	R	0201_0000h	<a href="#">18.4.35/723</a>

### 18.4.1 DCP Control Register 0 (DCP\_CTRL)

The CTRL register contains controls for the DCP module.

CTRL: 0x000



CTRL\_SET: 0x004

CTRL\_CLR: 0x008

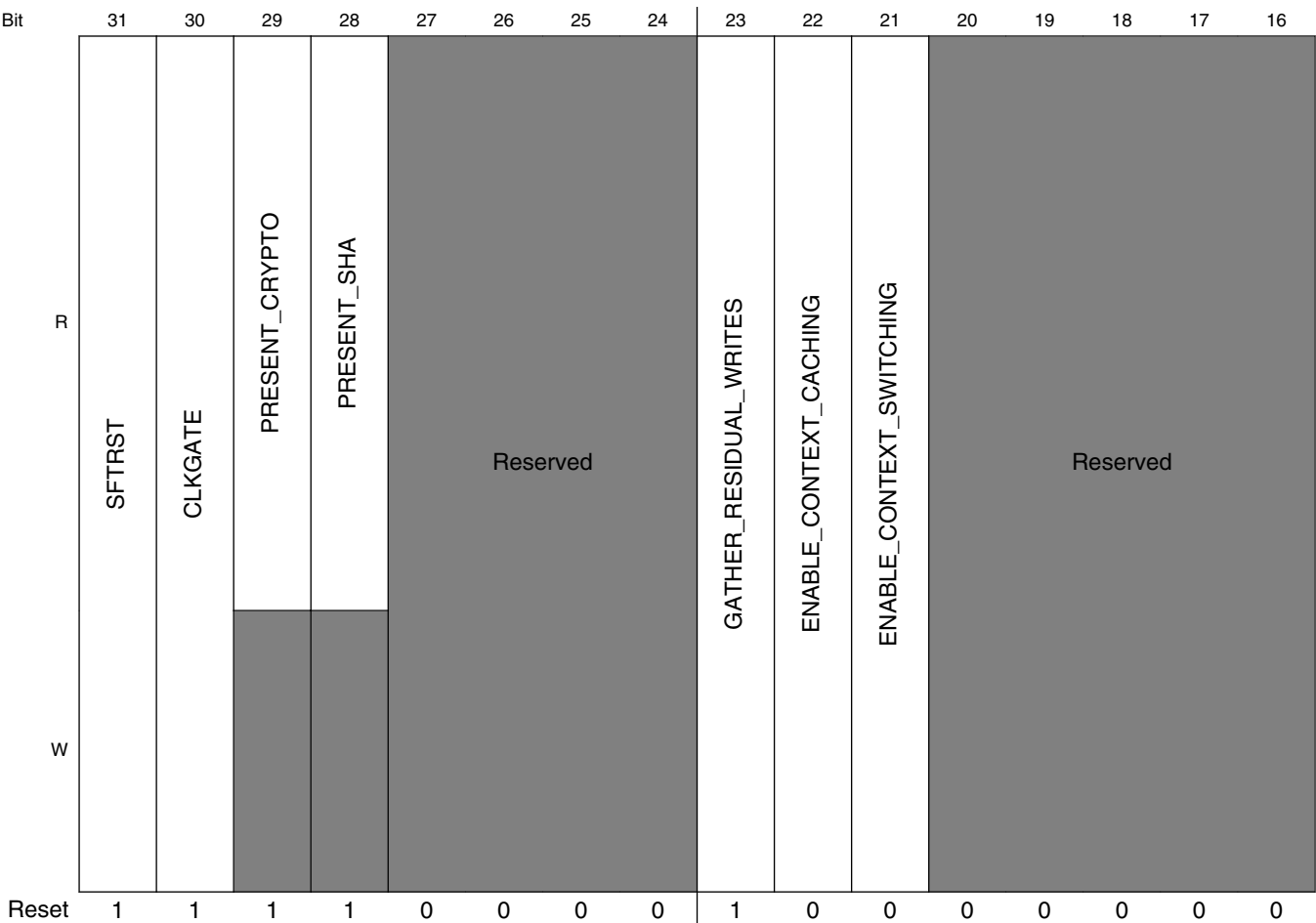
CTRL\_TOG: 0x00C

The Control register contains the primary controls for the DCP block. The present bits indicate which of the sub-features of the block are present in the hardware. The context control bits control how the DCP utilizes it's context buffer and the gather residual writes bit controls how the master handles writing misaligned data to the bus. Each channel and the color-space converter contains an independent interrupt enable.

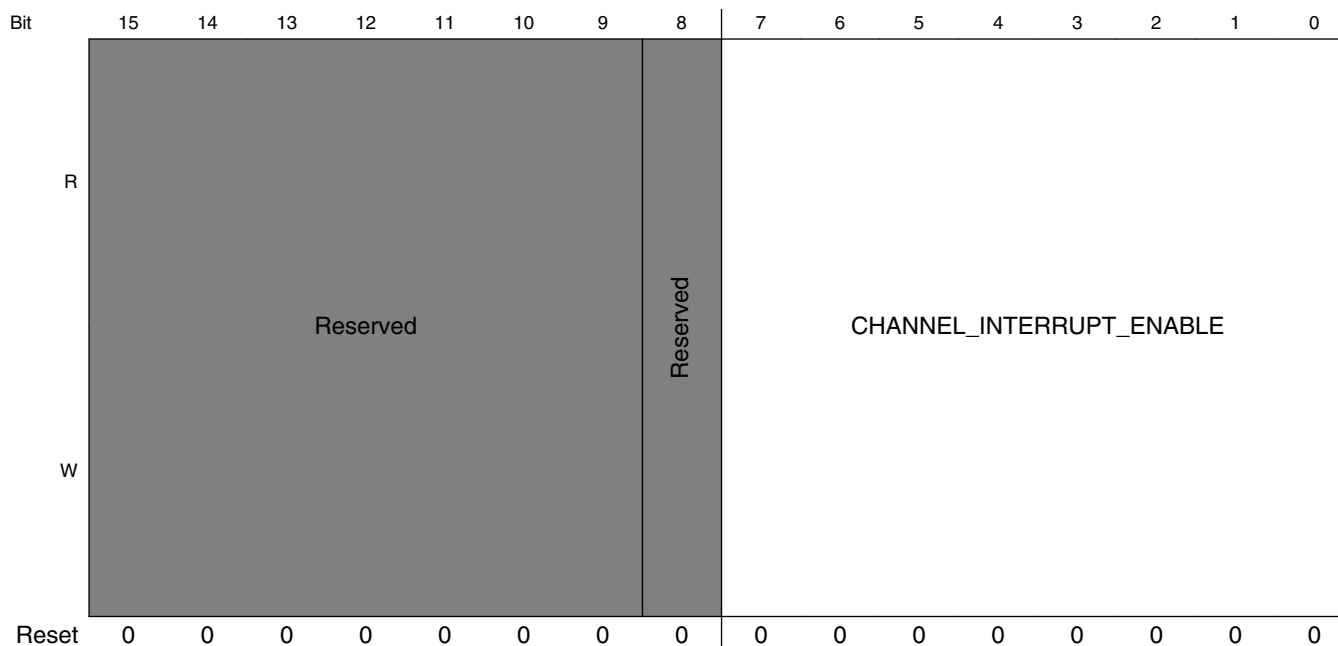
EXAMPLE

```
DCP_CTRL_SET(BM_DCP_CTRL_SFTRST);
DCP_CTRL_CLR(BM_DCP_CTRL_SFTRST | BM_DCP_CTRL_CLKGATE);
```

Address: 20F\_C000h base + 0h offset = 20F\_C000h



## DCP Memory Map/Register Definition



### DCP\_CTRL field descriptions

Field	Description
31 SFTRST	Set this bit to zero to enable normal DCP operation. Set this bit to one (default) to disable clocking with the DCP and hold it in its reset (lowest power) state. This bit can be turned on and then off to reset the DCP block to its default state.
30 CLKGATE	This bit must be set to zero for normal operation. When set to one it gates off the clocks to the block.
29 PRESENT_CRYPTO	Indicates whether the crypto (Cipher/Hash) functions are present. 0x1 <b>Present</b> — 0x0 <b>Absent</b> —
28 PRESENT_SHA	Indicates whether the SHA1/SHA2 functions are present. 0x1 <b>Present</b> — 0x0 <b>Absent</b> —
27–24 -	This field is reserved. Reserved, always set to zero.
23 GATHER_RESIDUAL_WRITES	Software should set this bit to enable ragged writes to unaligned buffers to be gathered between multiple write operations. This improves performance by removing several byte operations between write bursts. Trailing byte writes are held in a residual write data buffer and combined with a subsequent write to the buffer to form a word write.
22 ENABLE_CONTEXT_CACHING	Software should set this bit to enable caching of contexts between operations. If only a single channel is used for encryption/hashing, enabling caching causes the context to not be reloaded if the channel was the last to be used.
21 ENABLE_CONTEXT_SWITCHING	Enable automatic context switching for the channels. Software should set this bit if more than one channel is doing hashing or cipher operations that require context to be saved (for instance, when CBC mode is enabled). By disabling context switching, software can save the 208 bytes used for the context buffer.

Table continues on the next page...

**DCP\_CTRL field descriptions (continued)**

Field	Description
20–9 -	This field is reserved. Reserved, always set to zero.
8 RSVD_CSC_ INTERRUPT_ ENABLE	This field is reserved. Reserved, always set to zero.
7–0 CHANNEL_ INTERRUPT_ ENABLE	Per-channel interrupt enable bit. When set, the channel's interrupt will get routed to the interrupt controller. Channel 0 is routed to the dcp_vmi_irq signal and the other channels are combined (along with the CRC interrupt) into the dcp_irq signal.  0x01 <b>CH0</b> — 0x02 <b>CH1</b> — 0x04 <b>CH2</b> — 0x08 <b>CH3</b> —

**18.4.2 DCP Status Register (DCP\_STAT)**

The DCP Interrupt Status register provides channel interrupt status information.

STAT: 0x010

STAT\_SET: 0x014

STAT\_CLR: 0x018

STAT\_TOG: 0x01C

This register provides status feedback indicating the channel currently performing an operation and which channels have pending operations.

**EXAMPLE**

## DCP Memory Map/Register Definition

Address: 20F\_C000h base + 10h offset = 20F\_C010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			OTP_KEY_READY	CUR_CHANNEL				READY_CHANNELS							
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved							Reserved	Reserved				IRQ			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DCP\_STAT field descriptions**

Field	Description
31–29 -	This field is reserved. Reserved, always set to zero.
28 OTP_KEY_READY	When set, indicates that the OTP key has been shifted from the fuse block and is ready for use.
27–24 CUR_CHANNEL	Current (active) channel (encoded).  0x0 <b>None</b> — 0x1 <b>CH0</b> — 0x2 <b>CH1</b> — 0x3 <b>CH2</b> — 0x4 <b>CH3</b> —
23–16 READY_CHANNELS	Indicates which channels are ready to proceed with a transfer (active channel also included). Each bit is a one-hot indicating the request status for the associated channel.  0x01 <b>CH0</b> — 0x02 <b>CH1</b> — 0x04 <b>CH2</b> — 0x08 <b>CH3</b> —
15–9 -	This field is reserved. Reserved, always set to zero.
8 RSVD_IRQ	This field is reserved. Reserved, always set to zero.
7–4 -	This field is reserved. Reserved, always set to zero.
3–0 IRQ	Indicates which channels have pending interrupt requests. Channel 0's interrupt is routed through the dcp_vmi_irq and the other interrupt bits are routed through the dcp_irq.

**18.4.3 DCP Channel Control Register (DCP\_CHANNELCTRL)**

The DCP Channel Control register provides controls for channel arbitration and channel enables.

CHANNELCTRL: 0x020

CHANNELCTRL\_SET: 0x024

CHANNELCTRL\_CLR: 0x028

CHANNELCTRL\_TOG: 0x02C

This register provides status feedback indicating the channel currently performing an operation and which channels have pending operations.

**EXAMPLE**

## DCP Memory Map/Register Definition

```
BW_DCP_CHANNELCTRL_ENABLE_CHANNEL(BV_DCP_CHANNELCTRL_ENABLE_CHANNEL__CH0); //
enable channel 0
```

Address: 20F\_C000h base + 20h offset = 20F\_C020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															CH0_IRQ_MERGED
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HIGH_PRIORITY_CHANNEL								ENABLE_CHANNEL							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### DCP\_CHANNELCTRL field descriptions

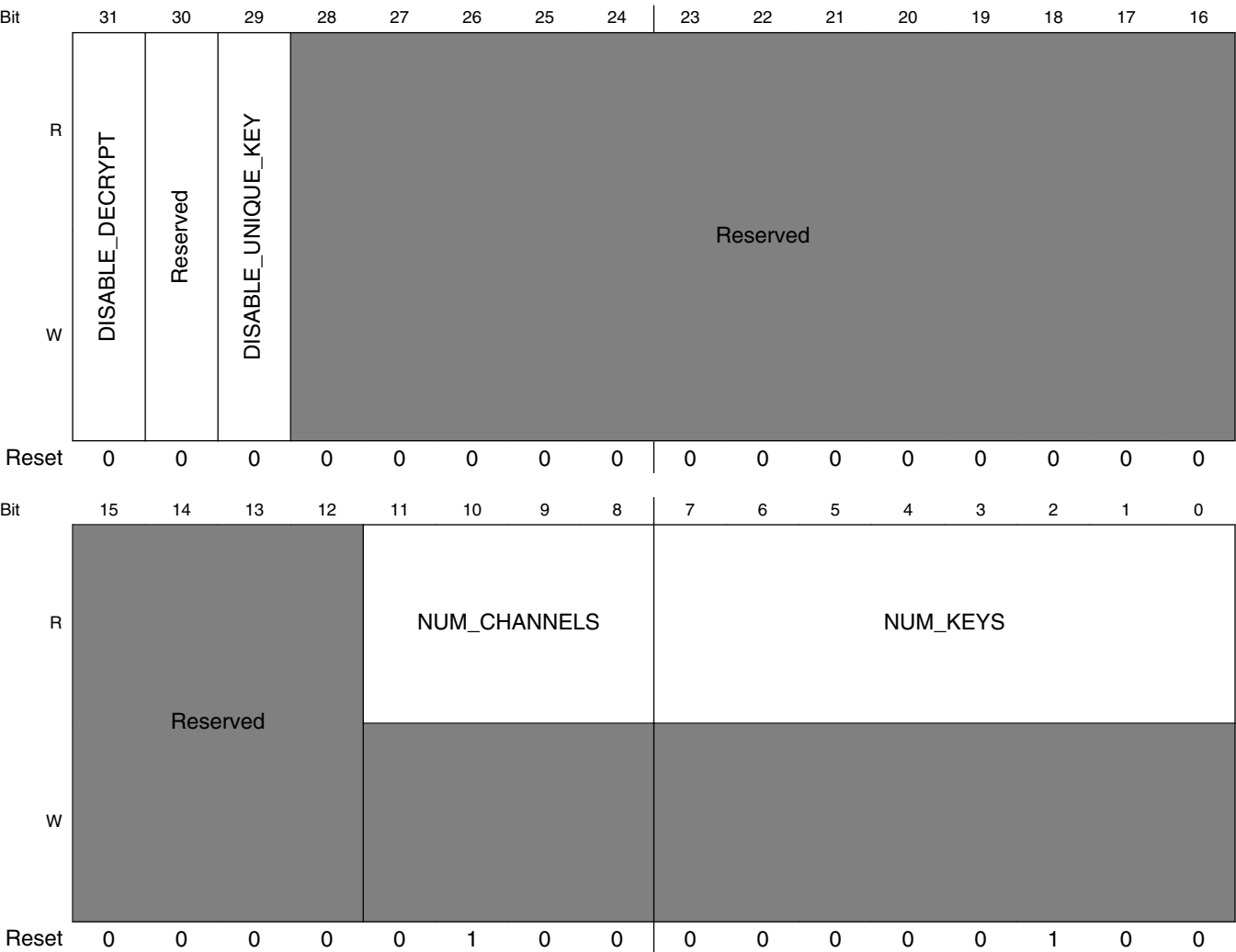
Field	Description
31–17 RSVD	This field is reserved. Reserved, always set to zero.
16 CH0_IRQ_MERGED	Indicates that the interrupt for channel 0 should be merged with the other interrupts on the shared dcp_irq interrupt. When set to 0, channel 0's interrupt will be routed to the separate dcp_vmi_irq. When set to 1, the interrupt will be routed to the shared DCP interrupt.
15–8 HIGH_PRIORITY_CHANNEL	Setting a bit in this field causes the corresponding channel to have high-priority arbitration. High priority channels will be arbitrated round-robin and will take precedence over other channels that are not marked as high priority.  0x01 <b>CH0</b> — 0x02 <b>CH1</b> — 0x04 <b>CH2</b> — 0x08 <b>CH3</b> —
7–0 ENABLE_CHANNEL	Setting a bit in this field will enabled the DMA channel associated with it. This field is a direct input to the DMA channel arbiter. When not enabled, the channel is denied access to the central DMA resources.  0x01 <b>CH0</b> — 0x02 <b>CH1</b> — 0x04 <b>CH2</b> — 0x08 <b>CH3</b> —

### 18.4.4 DCP Capability 0 Register (DCP\_CAPABILITY0)

This register contains additional information about the DCP module implementation parameters.

This register provides capability information for the DCP block. It indicates the number of channels implemented as well as the number of key storage locations available for software use.

Address: 20F\_C000h base + 30h offset = 20F\_C030h



DCP\_CAPABILITY0 field descriptions

Field	Description
31 DISABLE_DECRYPT	Write to a 1 to disable decryption. This bit can only be written by secure software and the value can only be cleared by a reset.

Table continues on the next page...

**DCP\_CAPABILITY0 field descriptions (continued)**

Field	Description
30 Reserved	This field is reserved. -
29 DISABLE_UNIQUE_KEY	Write to a 1 disable the per-device unique key. The device-specific hardware key may be selected by using a value of 0xFE in the key select field.
28–12 RSVD	This field is reserved. Reserved, always set to zero.
11–8 NUM_CHANNELS	Encoded value indicating the number of channels implemented in the design.
7–0 NUM_KEYS	Encoded value indicating the number of key storage locations implemented in the design.

**18.4.5 DCP Capability 1 Register (DCP\_CAPABILITY1)**

This register contains information about the algorithms available on the implementation.

This register provides capability information for the DCP block. It contains two fields indicating which encryption and hashing algorithms are present in the design. Each bit set indicates that support for the associated function is present.

Address: 20F\_C000h base + 40h offset = 20F\_C040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HASH_ALGORITHMS																CIPHER_ALGORITHMS															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

**DCP\_CAPABILITY1 field descriptions**

Field	Description
31–16 HASH_ALGORITHMS	One-hot field indicating which hashing features are implemented in HW. 0x0001 <b>SHA1</b> — 0x0002 <b>CRC32</b> — 0x0004 <b>SHA256</b> —
15–0 CIPHER_ALGORITHMS	One-hot field indicating which cipher algorithms are available. 0x0001 <b>AES128</b> —

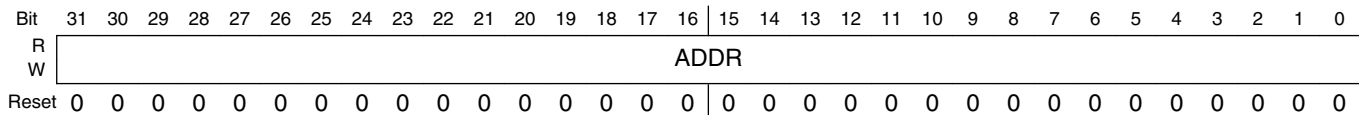


### 18.4.6 DCP Context Buffer Pointer (DCP\_CONTEXT)

This register contains a pointer to the memory region to be used for DCP context swap operations.

This register contains a pointer to the start of the context pointer memory in on-chip SRAM or off-chip SDRAM. This buffer will be used to store state information when the DCP module changes from one channel to another.

Address: 20F\_C000h base + 50h offset = 20F\_C050h



**DCP\_CONTEXT field descriptions**

Field	Description
31–0 ADDR	Context pointer address. Address should be located in system RAM and should be word-aligned for optimal performance.

### 18.4.7 DCP Key Index (DCP\_KEY)

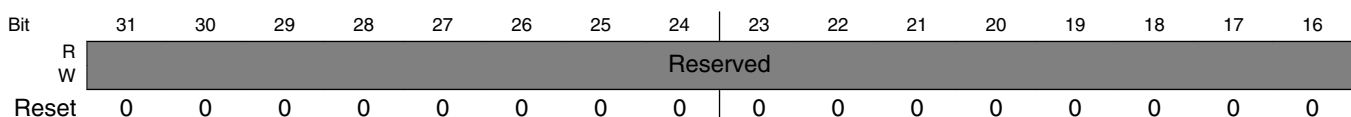
This register contains a pointer to the key location to be written.

The DCP module maintains a set of write-only keys that may be used by software. To write a key, software must first write the desired key index/subword to this register and then write the key values to the key registers (below). After each write to the key data register, the SUBWORD field will increment to allow software to write the subsequent key to be written without having to rewrite the key index.

#### EXAMPLE

```
// write key 0 to 0x00112233_44556677_8899aabb_ccddeeff
DCP_KEY_WR(BF_DCP_KEY_INDEX(0) | BF_DCP_KEY_SUBWORD(0)); // set key index to key 0, subword
0
DCP_KEYDATA_WR(0xccddeeff); // write key values (subword 0)
DCP_KEYDATA_WR(0x8899aabb); // write key values (subword 1)
DCP_KEYDATA_WR(0x44556677); // write key values (subword 2)
DCP_KEYDATA_WR(0x00112233); // write key values (subword 3)
```

Address: 20F\_C000h base + 60h offset = 20F\_C060h



## DCP Memory Map/Register Definition

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								Reserved		INDEX		Reserved		SUBWORD	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### DCP\_KEY field descriptions

Field	Description
31–8 RSVD	This field is reserved. Reserved, always set to zero.
7–6 RSVD_INDEX	This field is reserved. Reserved, always set to zero.
5–4 INDEX	Key index pointer. Valid indices are 0-[number_keys].
3–2 RSVD_SUBWORD	This field is reserved. Reserved, always set to zero.
1–0 SUBWORD	Key subword pointer. Valid indices are 0-3. After each write to the key data register, this field will increment.

## 18.4.8 DCP Key Data (DCP\_KEYDATA)

This register provides write access to the key/key subword specified by the Key Index Register.

Writing this location updates the selected subword for the key located at the index specified by the Key Index Register. A write also triggers the SUBWORD field of the KEY register to increment to the next higher word in the key.

### EXAMPLE

```
// write key 0 to 0x00112233_44556677_8899aabb_ccddeeff
DCP_KEY_WR(BF_DCP_KEY_INDEX(0) | BF_DCP_KEY_SUBWORD(0)); // set key index to key 0, subword
0
DCP_KEYDATA_WR(0xccddeeff); // write key values (subword 0)
DCP_KEYDATA_WR(0x8899aabb); // write key values (subword 1)
DCP_KEYDATA_WR(0x44556677); // write key values (subword 2)
DCP_KEYDATA_WR(0x00112233); // write key values (subword 3)
```

Address: 20F\_C000h base + 70h offset = 20F\_C070h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**DCP\_KEYDATA field descriptions**

Field	Description
31–0 DATA	Word 0 data for key. This is the least-significant word.

**18.4.9 DCP Work Packet 0 Status Register (DCP\_PACKET0)**

This register displays the values for the current work packet offset 0x00 (Next Command) field.

The Work Packet Status Registers show the contents of the currently executing packet. When the channels are inactive, the packet status register return 0. The register bits are fully documented here to document the packet structure in memory.

Address: 20F\_C000h base + 80h offset = 20F\_C080h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DCP\_PACKET0 field descriptions**

Field	Description
31–0 ADDR	Next Pointer Register,

**18.4.10 DCP Work Packet 1 Status Register (DCP\_PACKET1)**

This register displays the values for the current work packet offset 0x04 (control) field.

This register shows the contents of the Control0 register from the packet being processed.

DCP Memory Map/Register Definition

Address: 20F\_C000h base + 90h offset = 20F\_C090h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	TAG								OUTPUT_WORDSWAP	OUTPUT_BYTESWAP	INPUT_WORDSWAP	INPUT_BYTESWAP	KEY_WORDSWAP	KEY_BYTESWAP	TEST_SEMA_IRQ	CONSTANT_FILL
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HASH_OUTPUT	CHECK_HASH	HASH_TERM	HASH_INIT	PAYLOAD_KEY	OTP_KEY	CIPHER_INIT	CIPHER_ENCRYPT	ENABLE_BLIT	ENABLE_HASH	ENABLE_CIPHER	ENABLE_MEMCOPY	CHAIN_CONTIGUOUS	CHAIN	DECR_SEMAPHORE	INTERRUPT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

DCP\_PACKET1 field descriptions

Field	Description
31–24 TAG	Packet Tag
23 OUTPUT_ WORDSWAP	Reflects whether the DCP engine will wordswap output data (big-endian data).
22 OUTPUT_ BYTESWAP	Reflects whether the DCP engine will byteswap output data (big-endian data).
21 INPUT_ WORDSWAP	Reflects whether the DCP engine will wordswap input data (big-endian data).
20 INPUT_ BYTESWAP	Reflects whether the DCP engine will byteswap input data (big-endian data).
19 KEY_ WORDSWAP	Reflects whether the DCP engine will swap key words (big-endian key).
18 KEY_ BYTESWAP	Reflects whether the DCP engine will swap key bytes (big-endian key).

Table continues on the next page...

## DCP\_PACKET1 field descriptions (continued)

Field	Description
17 TEST_SEMA_ IRQ	This bit is used to test the channel semaphore transition to 0. FOR TEST USE ONLY!
16 CONSTANT_ FILL	When this bit is set (MEMCOPY and BLIT modes only), the DCP will simply fill the destination buffer with the value found in the Source Address field.
15 HASH_OUTPUT	When hashing is enabled, this bit controls whether the input or output data is hashed. 0 <b>INPUT</b> — 1 <b>OUTPUT</b> —
14 CHECK_HASH	Reflects whether the calculated hash value should be compared against the hash provided in the payload.
13 HASH_TERM	Reflects whether the current hashing block is the final block in the hashing operation, so the hash padding should be applied by hardware.
12 HASH_INIT	Reflects whether the current hashing block is the initial block in the hashing operation, so the hash registers should be initialized before the operation.
11 PAYLOAD_KEY	When set, indicates the payload contains the key. This bit takes precedence over the OTP_KEY control
10 OTP_KEY	Reflects whether a hardware-based key should be used. The KEY_SELECT field from the Control1 field is used to select from multiple hardware keys. The PAYLOAD_KEY bit takes precedence over the OTP_KEY bit.
9 CIPHER_INIT	Reflects whether the cipher block should load the initialization vector from the payload for this operation.
8 CIPHER_ ENCRYPT	When the cipher block is enabled, this bit indicates whether the operation is encryption or decryption. 1 <b>ENCRYPT</b> — 0 <b>DECRYPT</b> —
7 ENABLE_BLIT	Reflects whether the DCP should perform a blit operation. Source data is always continuous and the destination buffer is written in run/stride format. When set, the BUFFER_SIZE field is treated as two 16-bit values for the X-Y extents of the blit operation.
6 ENABLE_HASH	Reflects whether the selected hashing function should be enabled for this operation.
5 ENABLE_ CIPHER	Reflects whether the selected cipher function should be enabled for this operation.
4 ENABLE_ MEMCOPY	Reflects whether the selected hashing function should be enabled for this operation.
3 CHAIN_ CONTIGUOUS	Reflects whether the next packet's address is located following this packet's payload.
2 CHAIN	Reflects whether the next command pointer register should be loaded into the channel's current descriptor pointer.
1 DECR_ SEMAPHORE	Reflects whether the channel's semaphore should be decremented at the end of the current operation. When the semaphore reaches a value of zero, no more operations will be issued from the channel.

*Table continues on the next page...*

**DCP\_PACKET1 field descriptions (continued)**

Field	Description
0 INTERRUPT	Reflects whether the channel should issue an interrupt upon completion of the packet.

**18.4.11 DCP Work Packet 2 Status Register (DCP\_PACKET2)**

This register displays the values for the current work packet offset 0x08 (Control1) field.

This register shows the contents of the Control0 register from the packet being processed.

Address: 20F\_C000h base + A0h offset = 20F\_C0A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	CIPHER_CFG								Reserved				HASH_SELECT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	KEY_SELECT								CIPHER_MODE				CIPHER_SELECT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DCP\_PACKET2 field descriptions**

Field	Description
31–24 CIPHER_CFG	Cipher configuration bits. Optional configuration bits required for ciphers
23–20 RSVD	This field is reserved. Reserved, always set to zero.
19–16 HASH_SELECT	Hash Selection Field 0x00 <b>SHA1</b> — 0x01 <b>CRC32</b> — 0x02 <b>SHA256</b> —
15–8 KEY_SELECT	Key Selection Field. The value here reflects the key index for the cipher operation. Values 0-3 refer to the software keys that can be written to the key RAM. The OTP key or the unique device-specific key may also be selected with a value of 0xFF (OTP key) or 0xFE (unique key). 0x00 <b>KEY0</b> — 0x01 <b>KEY1</b> — 0x02 <b>KEY2</b> — 0x03 <b>KEY3</b> — 0xFE <b>UNIQUE_KEY</b> — 0xFF <b>OTP_KEY</b> —
7–4 CIPHER_MODE	Cipher Mode Selection Field. Reflects the mode of operation for cipher operations.

Table continues on the next page...

**DCP\_PACKET2 field descriptions (continued)**

Field	Description
	0x00 <b>ECB</b> — 0x01 <b>CBC</b> —
3–0 CIPHER_SELECT	Cipher Selection Field 0x00 <b>AES128</b> —

**18.4.12 DCP Work Packet 3 Status Register (DCP\_PACKET3)**

This register displays the values for the current work packet offset 0x0C (Source Address) field.

This register shows the contents of the Source Address register from the packet being processed. When the **CONSTANT\_FILL** bit in the Control 0 field is set, this field contains the data written to the destination buffer.

Address: 20F\_C000h base + B0h offset = 20F\_C0B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DCP\_PACKET3 field descriptions**

Field	Description
31–0 ADDR	Source Buffer Address Pointer. This value is the working value and will update as the operation proceeds.

**18.4.13 DCP Work Packet 4 Status Register (DCP\_PACKET4)**

This register displays the values for the current work packet offset 0x10 (Destination Address) field.

This register shows the contents of the Destination Address register from the packet being processed.

Address: 20F\_C000h base + C0h offset = 20F\_C0C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



**DCP\_PACKET4 field descriptions**

Field	Description
31–0 ADDR	Destination Buffer Address Pointer. This value is the working value and will update as the operation proceeds.

**18.4.14 DCP Work Packet 5 Status Register (DCP\_PACKET5)**

This register displays the values for the current work packet offset 0x14 (Buffer Size) field.

This register shows the contents of the bytecount register from the packet being processed. The field can be considered either a byte count or a buffer size. The logic treats this as a decrementing count of bytes from the buffer size programmed into the field. As the transaction proceeds, the logic will decrement the bytecount as data is written to the destination buffer. For blit operations, the top 16-bits of this field represents the number of lines (y size) in the blit and the lower 16-bits represent the number of bytes in a line (x size).

Address: 20F\_C000h base + D0h offset = 20F\_C0D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COUNT																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DCP\_PACKET5 field descriptions**

Field	Description
31–0 COUNT	Byte Count register. This value is the working value and will update as the operation proceeds.

**18.4.15 DCP Work Packet 6 Status Register (DCP\_PACKET6)**

This register displays the values for the current work packet offset 0x1C (Payload Pointer) field.

This register shows the contents of the payload pointer fieldr from the packet being processed.

## DCP Memory Map/Register Definition

Address: 20F\_C000h base + E0h offset = 20F\_C0E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### DCP\_PACKET6 field descriptions

Field	Description
31–0 ADDR	This register reflects the payload pointer for the current control packet.

## 18.4.16 DCP Channel 0 Command Pointer Address Register (DCP\_CH0CMDPTR)

The DCP channel 0 current command address register points to the multiword descriptor that is to be executed (or currently being executed). The channel may be activated by writing the command pointer address to a valid descriptor in memory and then updating the semaphore to a non-zero value. After the engine completes processing of a descriptor, the "next\_ptr" field from the descriptor is moved into this register to enable processing of the next descriptor. All channels with a non-zero semaphore value will arbitrate for access to the engine for the subsequent operation.

DCP Channel 0 is controlled by a variable sized command structure. This register points to the command structure to be executed.

### EXAMPLE

```
DCP_CHnCMDPTR_WR(0, v);    // Write channel 0 command pointer
pCurptr = (DCP_chncmdptr_t *) DCP_CHnCMDPTR_RD(0); // Read current command
pointer
```

Address: 20F\_C000h base + 100h offset = 20F\_C100h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### DCP\_CH0CMDPTR field descriptions

Field	Description
31–0 ADDR	Pointer to descriptor structure to be processed for channel 0.

### 18.4.17 DCP Channel 0 Semaphore Register (DCP\_CH0SEMA)

The DCP Channel 0 semaphore register is used to synchronize the ARM platform instruction stream and the DMA chain processing state. After a command chain has been generated in memory, software should write the address of the first command descriptor to the CMDPTR register and then write a non-zero value to the semaphore register to indicate that the channel is active. Each command packet has a chaining bit which indicates that another descriptor should be loaded into the channel upon completion of the current descriptor. If the chaining bit is not set, the next address will not be loaded into the CMDPTR register. Each packet also contains a "decrement semaphore" bit, which indicates that the counting semaphore should be decremented after the operation. A channel is considered active when the semaphore is a non-zero value. When programming a series operations, software must properly program the semaphore values in conjunction with the "decrement\_semaphore" bits in the control packets to ensure that the proper number of descriptors are activated. A semaphore may be cleared by software by writing 0xFF to the DCP\_CHnSEMA\_CLR register. The logic will also clear the semaphore if an error has occurred.

Each DCP channel has an 8 bit counting semaphore that is used to synchronize between the program stream and the DCP chain processing. After processing each control packet, the DCP decrements the semaphore if it is non-zero. The channel will continue processing packets as long as the semaphore contains a non-zero value and the CHAIN or CHAIN\_CONTIGUOUS control bits in the Control0 field are set.

Address: 20F\_C000h base + 110h offset = 20F\_C110h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								VALUE								Reserved								INCREMENT							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**DCP\_CH0SEMA field descriptions**

Field	Description
31–24 -	This field is reserved. Reserved, always set to zero.
23–16 VALUE	This read-only field shows the current (instantaneous) value of the semaphore counter.
15–8 -	This field is reserved. Reserved, always set to zero.
7–0 INCREMENT	The value written to this field is added to the semaphore count in an atomic way such that simultaneous software adds and DCP hardware subtracts happening on the same clock are protected. This bit field reads back a value of 0x00. Writing a value of 0x02 increments the semaphore count by two, unless the DCP channel decrements the count on the same clock, then the count is incremented by a net one. The semaphore may be cleared by writing 0xFF to the DCP_CHnSEMA_CLR register.

18.4.18 DCP Channel 0 Status Register (DCP\_CH0STAT)

The DCP Channel 0 Interrupt Status register contains the interrupt status bit and the tag of the last completed operation from the command chain. If an error occurs during processing, the ERROR bit is set and an interrupt is generated.

DCP\_CH0STAT: 0x120

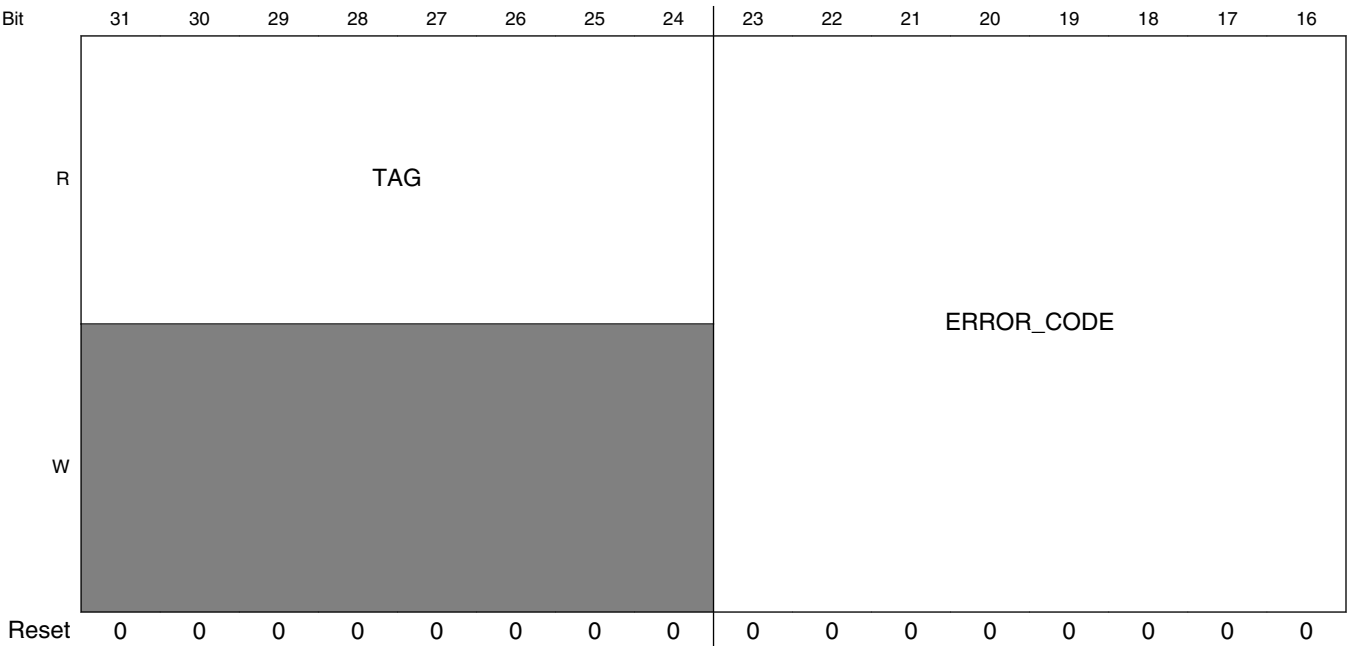
CH0STAT\_SET: 0x124

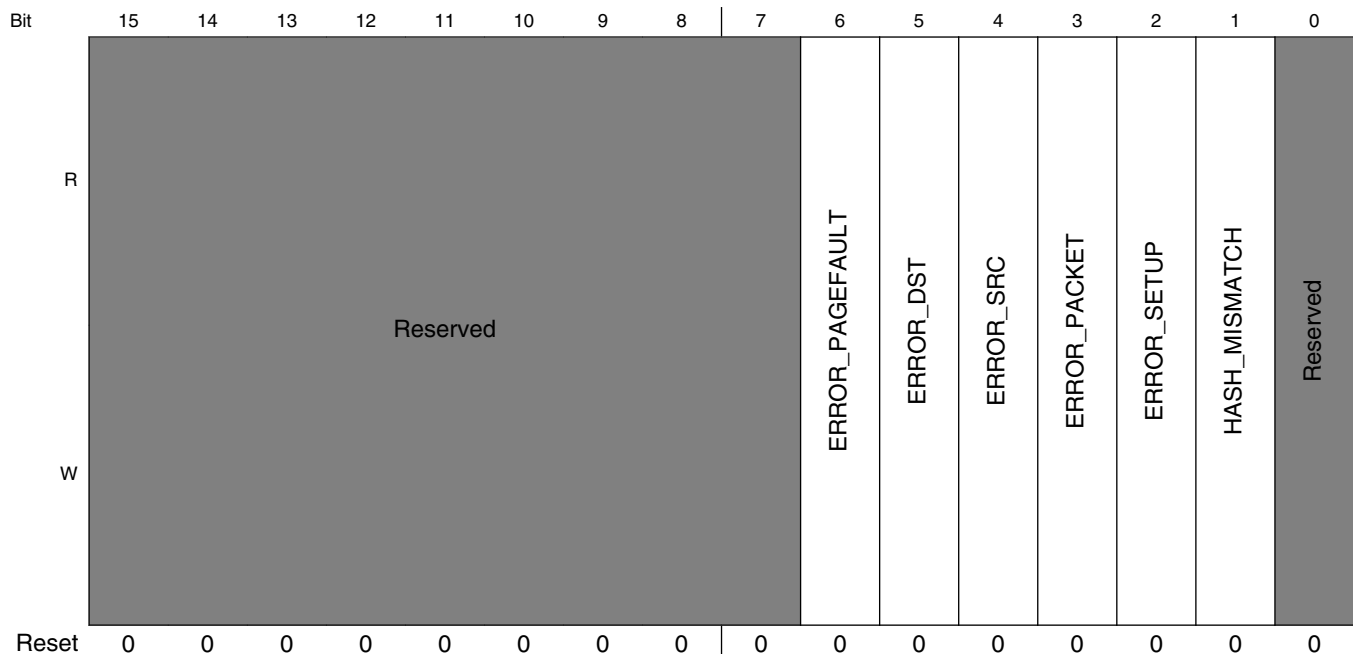
CH0STAT\_CLR: 0x128

CH0STAT\_TOG: 0x12C

The interrupt status register is updated at the end of each work packet. If the interrupt bit is set in the command packet's command field, an interrupt will be generated once the packet has completed. In addition, the tag value from the command is stored in the TAG field so that software can identify which command structure was the last to complete. If an error occurs, the ERROR bit is set and processing of the command chain is halted.

Address: 20F\_C000h base + 120h offset = 20F\_C120h





DCP\_CH0STAT field descriptions

Field	Description
31–24 TAG	Indicates the tag from the last completed packet in the command structure
23–16 ERROR_CODE	Indicates additional error codes for some error conditions. 0x01 <b>NEXT_CHAIN_IS_0</b> — Error signalled because the next pointer is 0x00000000 0x02 <b>NO_CHAIN</b> — Error signalled because the semaphore is nonzero and neither chain bit is set 0x03 <b>CONTEXT_ERROR</b> — Error signalled because an error was reported reading/writing the context buffer 0x04 <b>PAYLOAD_ERROR</b> — Error signalled because an error was reported reading/writing the payload 0x05 <b>INVALID_MODE</b> — Error signalled because the control packet specifies an invalid mode select (for instance, blit + hash)
15–7 -	This field is reserved. Reserved, always set to zero.
6 ERROR_PAGEFAULT	This bit indicates a page fault occurred while converting a virtual address to a physical address.. When an error is detected, the channel's processing will stop until the error handled by software.
5 ERROR_DST	This bit indicates a bus error occurred when storing to the destination buffer. When an error is detected, the channel's processing will stop until the error handled by software.
4 ERROR_SRC	This bit indicates a bus error occurred when reading from the source buffer. When an error is detected, the channel's processing will stop until the error handled by software.
3 ERROR_PACKET	This bit indicates that a a bus error occurred when reading the packet or payload or when writing status back to the packet payload. When an error is detected, the channel's processing will stop until the error is handled by software.
2 ERROR_SETUP	This bit indicates that the hardware has detected an invalid programming configuration such as a buffer length that is not a multiple of the natural data size for the operation. When an error is detected, the channel's processing will stop until the error is handled by software.

Table continues on the next page...

**DCP\_CH0STAT field descriptions (continued)**

Field	Description
1 HASH_ MISMATCH	The bit indicates that a hashing check operation mismatched for control packets that enable the HASH_CHECK bit. When an error is detected, the channel's processing will stop until the error is handled by software.
0 RSVD_ COMPLETE	This field is reserved. This bit will always read 0 in the status register, but will be set to 1 in the packet status field after processing of the packet has completed. This was done so that software can verify that each packet completed properly in a chain of commands for cases when an interrupt is issued only for the last item in a packet. The completion bit for the channel is effectively the channel interrupt status bit.

**18.4.19 DCP Channel 0 Options Register (DCP\_CH0OPTS)**

The DCP Channel 0 Options Status register contains optional control information that may be used to further tune the behavior of the channel.

DCP\_CH0OPTS: 0x130

CH0OPTS\_SET: 0x134

CH0OPTS\_CLR: 0x138

CH0OPTS\_TOG: 0x13C

The options register can be used to control optional features of the channels.

Address: 20F\_C000h base + 130h offset = 20F\_C130h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																RECOVERY_TIMER															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**DCP\_CH0OPTS field descriptions**

Field	Description
31–16 RSVD	This field is reserved. Reserved, always set to zero.
15–0 RECOVERY_ TIMER	This field indicates the recovery time for the channel. After each operation, the recover timer for the channel is initialized with this value and then decremented until the timer reaches zero. The channel will not initiate another operation for the next packet in the chain until the recovery time has been satisfied. The timebase for the recovery timer is 16 HCLK clock cycles, providing a range of 0ns to 8.3ms at 133 MHz operation.

### 18.4.20 DCP Channel 1 Command Pointer Address Register (DCP\_CH1CMDPTR)

The DCP channel 1 current command address register points to the multiword descriptor that is to be executed (or currently being executed). The channel may be activated by writing the command pointer address to a valid descriptor in memory and then updating the semaphore to a non-zero value. After the engine completes processing of a descriptor, the "next\_ptr" field from the descriptor is moved into this register to enable processing of the next descriptor. All channels with a non-zero semaphore value will arbitrate for access to the engine for the subsequent operation.

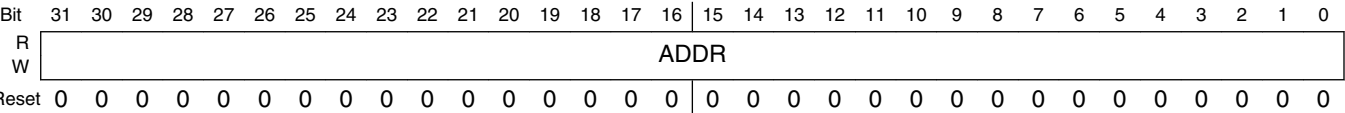
DCP Channel 1 is controlled by a variable sized command structure. This register points to the command structure to be executed.

#### EXAMPLE

```

DCP_CHn_CMDPTR_WR(1, v); // Write channel 1 command pointer
pCurptr = (DCP_chn_cmdptr_t *) DCP_CHn_CMDPTR_RD(1); // Read current command
pointer
    
```

Address: 20F\_C000h base + 140h offset = 20F\_C140h



DCP\_CH1CMDPTR field descriptions

Field	Description
31–0 ADDR	Pointer to descriptor structure to be processed for channel 1.

## 18.4.21 DCP Channel 1 Semaphore Register (DCP\_CH1SEMA)

The DCP Channel 1 semaphore register is used to synchronize the ARM platform instruction stream and the DMA chain processing state. After a command chain has been generated in memory, software should write the address of the first command descriptor to the CMDPTR register and then write a non-zero value to the semaphore register to indicate that the channel is active. Each command packet has a chaining bit which indicates that another descriptor should be loaded into the channel upon completion of the current descriptor. If the chaining bit is not set, the next address will not be loaded into the CMDPTR register. Each packet also contains a "decrement semaphore" bit, which indicates that the counting semaphore should be decremented after the operation. A channel is considered active when the semaphore is a non-zero value. When programming a series operations, software must properly program the semaphore values in conjunction with the "decrement\_semaphore" bits in the control packets to ensure that the proper number of descriptors are activated. A semaphore may be cleared by software by writing 0xFF to the DCP\_CHnSEMA\_CLR register. The logic will also clear the semaphore if an error has occurred.

Each DCP channel has an 8 bit counting semaphore that is used to synchronize between the program stream and the DCP chain processing. DCP processing continues until the engine attempts to decrement a semaphore that has already reached a value of zero. When the attempt is made, the DCP channel is stalled until software increments the semaphore count.

Address: 20F\_C000h base + 150h offset = 20F\_C150h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								VALUE								Reserved								INCREMENT							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DCP\_CH1SEMA field descriptions**

Field	Description
31–24 -	This field is reserved. Reserved, always set to zero.
23–16 VALUE	This read-only field shows the current (instantaneous) value of the semaphore counter.
15–8 -	This field is reserved. Reserved, always set to zero.
7–0 INCREMENT	The value written to this field is added to the semaphore count in an atomic way such that simultaneous software adds and DCP hardware subtracts happening on the same clock are protected. This bit field reads back a value of 0x00. Writing a value of 0x02 increments the semaphore count by two, unless the DCP channel decrements the count on the same clock, then the count is incremented by a net one. The semaphore may be cleared by writing 0xFF to the DCP_CHnSEMA_CLR register.



### 18.4.22 DCP Channel 1 Status Register (DCP\_CH1STAT)

The DCP Channel 1 Interrupt Status register contains the interrupt status bit and the tag of the last completed operation from the command chain. If an error occurs during processing, the ERROR bit is set and an interrupt is generated.

CH1STAT: 0x160

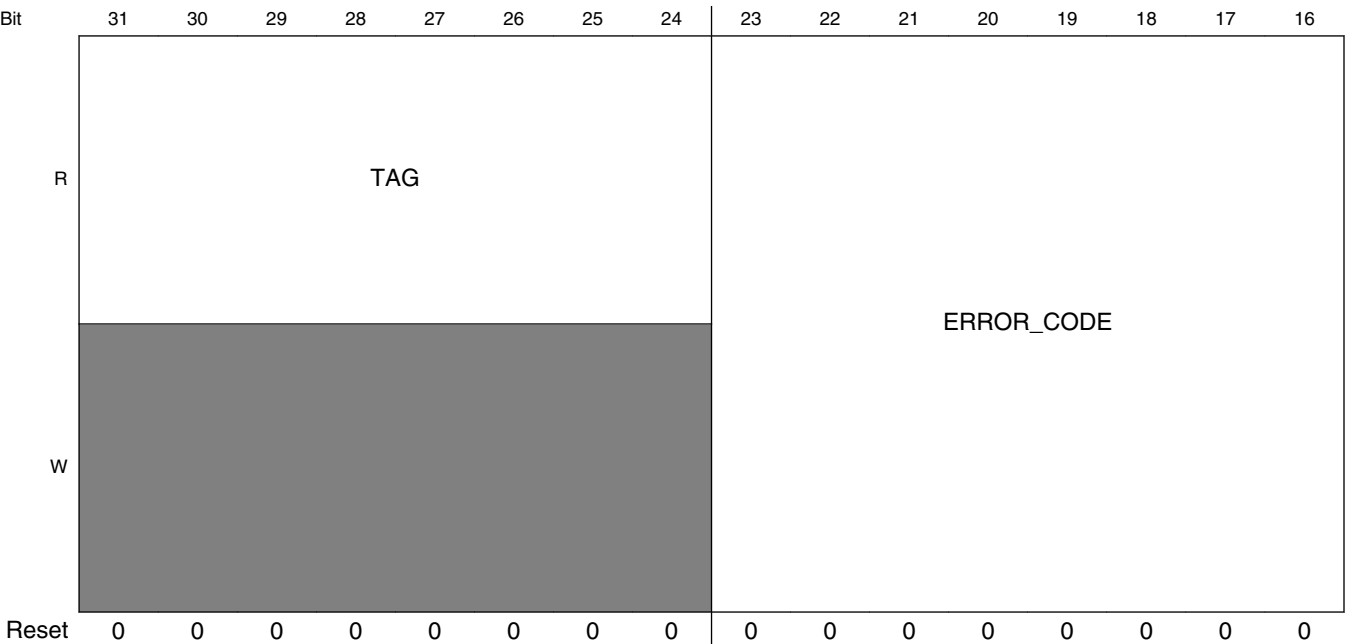
CH1STAT\_SET: 0x164

CH1STAT\_CLR: 0x168

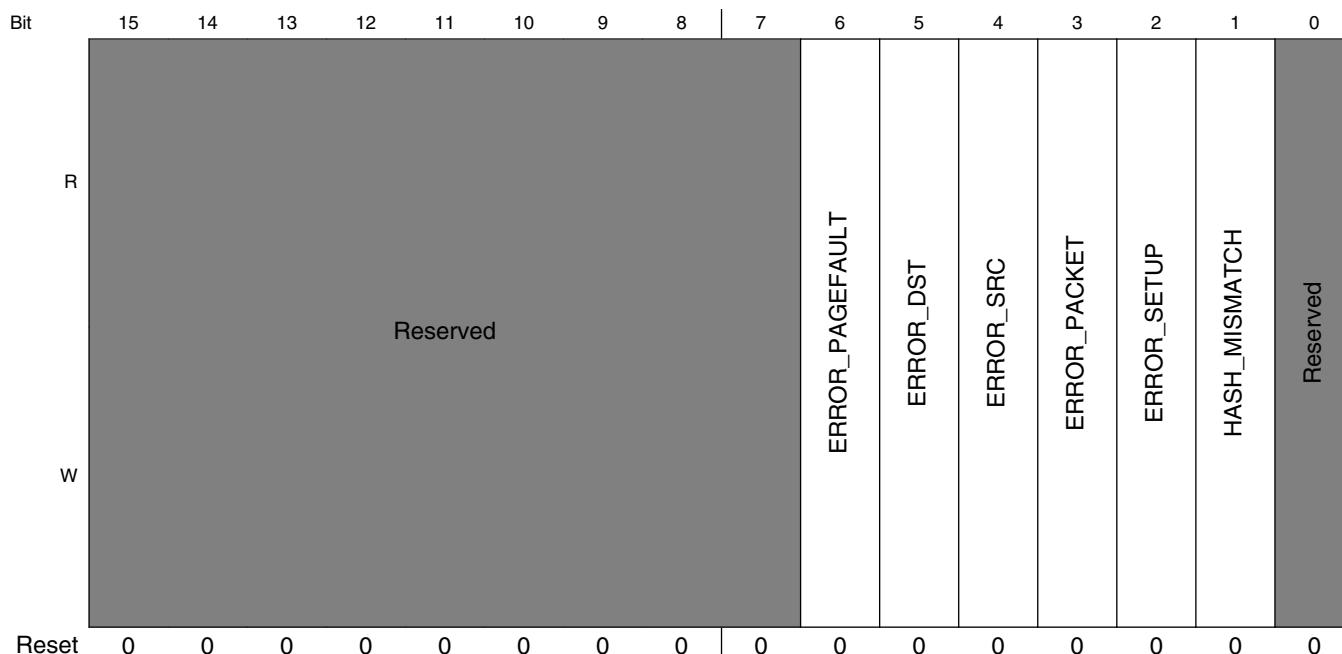
CH1STAT\_TOG: 0x16C

The interrupt status register is updated at the end of each work packet. If the interrupt bit is set in the command packet's command field, an interrupt will be generated once the packet has completed. In addition, the tag value from the command is stored in the TAG field so that software can identify which command structure was the last to complete. If an error occurs, the ERROR bit is set and processing of the command chain is halted.

Address: 20F\_C000h base + 160h offset = 20F\_C160h



## DCP Memory Map/Register Definition



### DCP\_CH1STAT field descriptions

Field	Description
31–24 TAG	Indicates the tag from the last completed packet in the command structure
23–16 ERROR_CODE	Indicates additional error codes for some error conditions. 0x01 <b>NEXT_CHAIN_IS_0</b> — Error signalled because the next pointer is 0x00000000 0x02 <b>NO_CHAIN</b> — Error signalled because the semaphore is nonzero and neither chain bit is set 0x03 <b>CONTEXT_ERROR</b> — Error signalled because an error was reported reading/writing the context buffer 0x04 <b>PAYLOAD_ERROR</b> — Error signalled because an error was reported reading/writing the payload 0x05 <b>INVALID_MODE</b> — Error signalled because the control packet specifies an invalid mode select (for instance, blit + hash)
15–7 -	This field is reserved. Reserved, always set to zero.
6 ERROR_PAGEFAULT	This bit indicates a page fault occurred while converting a virtual address to a physical address.. When an error is detected, the channel's processing will stop until the error handled by software.
5 ERROR_DST	This bit indicates a bus error occurred when storing to the destination buffer. When an error is detected, the channel's processing will stop until the error handled by software.
4 ERROR_SRC	This bit indicates a bus error occurred when reading from the source buffer. When an error is detected, the channel's processing will stop until the error handled by software.
3 ERROR_PACKET	This bit indicates that a bus error occurs when reading the packet or payload or when writing status back to the packet payload. When an error is detected, the channel's processing will stop until the error is handled by software.
2 ERROR_SETUP	This bit indicates that the hardware detected an invalid programming configuration such as a buffer length that is not a multiple of the natural data size for the operation. When an error is detected, the channel's processing will stop until the error is handled by software.

Table continues on the next page...

**DCP\_CH1STAT field descriptions (continued)**

Field	Description
1 HASH_ MISMATCH	The bit indicates that a hashing check operation mismatched for control packets that enable the HASH_CHECK bit. When an error is detected, the channel's processing will stop until the error is handled by software.
0 RSVD_ COMPLETE	This field is reserved. This bit will always read 0 in the status register, but will be set to 1 in the packet status field after processing of the packet has completed. This was done so that software can verify that each packet completed properly in a chain of commands for cases when an interrupt is issued only for the last item in a packet. The completion bit for the channel is effectively the channel interrupt status bit.

**18.4.23 DCP Channel 1 Options Register (DCP\_CH1OPTS)**

The DCP Channel 1 Options Status register contains optional control information that may be used to further tune the behavior of the channel.

DCP\_CH1OPTS: 0x170

DCP\_CH1OPTS\_SET: 0x174

CH1OPTS\_CLR: 0x178

CH1OPTS\_TOG: 0x17C

The options register can be used to control optional features of the channels.

Address: 20F\_C000h base + 170h offset = 20F\_C170h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																RECOVERY_TIMER															
W	Reserved																RECOVERY_TIMER															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**DCP\_CH1OPTS field descriptions**

Field	Description
31–16 RSVD	This field is reserved. Reserved, always set to zero.
15–0 RECOVERY_ TIMER	This field indicates the recovery time for the channel. After each operation, the recover timer for the channel is initialized with this value and then decremented until the timer reaches zero. The channel will not initiate operation on the next packet in the chain until the recovery time has been satisfied. The timebase for the recovery timer is 16 HCLK clock cycles, providing a range of 0ns to 8.3ms at 133 MHz operation.

### 18.4.24 DCP Channel 2 Command Pointer Address Register (DCP\_CH2CMDPTR)

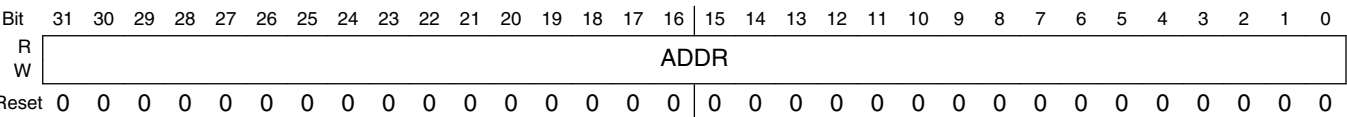
The DCP channel 2 current command address register points to the multiword descriptor that is to be executed (or currently being executed). The channel may be activated by writing the command pointer address to a valid descriptor in memory and then updating the semaphore to a non-zero value. After the engine completes processing of a descriptor, the "next\_ptr" field from the descriptor is moved into this register to enable processing of the next descriptor. All channels with a non-zero semaphore value will arbitrate for access to the engine for the subsequent operation.

DCP Channel 2 is controlled by a variable sized command structure. This register points to the command structure to be executed.

EXAMPLE

```
DCP_CHn_CMDPTR_WR(2, v); // Write channel 2 command pointer
pCurptr = (DCP_chn_cmdptr_t *) DCP_CHn_CMDPTR_RD(2); // Read current command
pointer
```

Address: 20F\_C000h base + 180h offset = 20F\_C180h



DCP\_CH2CMDPTR field descriptions

Field	Description
31–0 ADDR	Pointer to descriptor structure to be processed for channel 2.

### 18.4.25 DCP Channel 2 Semaphore Register (DCP\_CH2SEMA)

The DCP Channel 2 semaphore register is used to synchronize the ARM platform instruction stream and the DMA chain processing state. After a command chain has been generated in memory, software should write the address of the first command descriptor to the CMDPTR register and then write a non-zero value to the semaphore register to indicate that the channel is active. Each command packet has a chaining bit which indicates that another descriptor should be loaded into the channel upon completion of the current descriptor. If the chaining bit is not set, the next address will not be loaded into the CMDPTR register. Each packet also contains a "decrement semaphore" bit, which indicates that the counting semaphore should be decremented after the operation. A channel is considered active when the semaphore is a non-zero value. When programming a series operations, software must properly program the semaphore values in conjunction with the "decrement\_semaphore" bits in the control packets to ensure that the proper number of descriptors are activated. A semaphore may be cleared by software by writing 0xFF to the DCP\_CHnSEMA\_CLR register. The logic will also clear the semaphore if an error has occurred.

Each DCP channel has an 8 bit counting semaphore that is used to synchronize between the program stream and the DCP chain processing. DCP processing continues until the engine attempts to decrement a semaphore that has already reached a value of zero. When the attempt is made, the DCP channel is stalled until software increments the semaphore count.

Address: 20F\_C000h base + 190h offset = 20F\_C190h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								VALUE								Reserved								INCREMENT							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DCP\_CH2SEMA field descriptions**

Field	Description
31–24 -	This field is reserved. Reserved, always set to zero.
23–16 VALUE	This read-only field shows the current (instantaneous) value of the semaphore counter.
15–8 -	This field is reserved. Reserved, always set to zero.
7–0 INCREMENT	The value written to this field is added to the semaphore count in an atomic way such that simultaneous software adds and DCP hardware subtracts happening on the same clock are protected. This bit field reads back a value of 0x00. Writing a value of 0x02 increments the semaphore count by two, unless the DCP channel decrements the count on the same clock, then the count is incremented by a net one. The semaphore may be cleared by writing 0xFF to the DCP_CHnSEMA_CLR register.

18.4.26 DCP Channel 2 Status Register (DCP\_CH2STAT)

The DCP Channel 2 Interrupt Status register contains the interrupt status bit and the tag of the last completed operation from the command chain. If an error occurs during processing, the ERROR bit is set and an interrupt is generated.

CH2STAT: 0x1A0

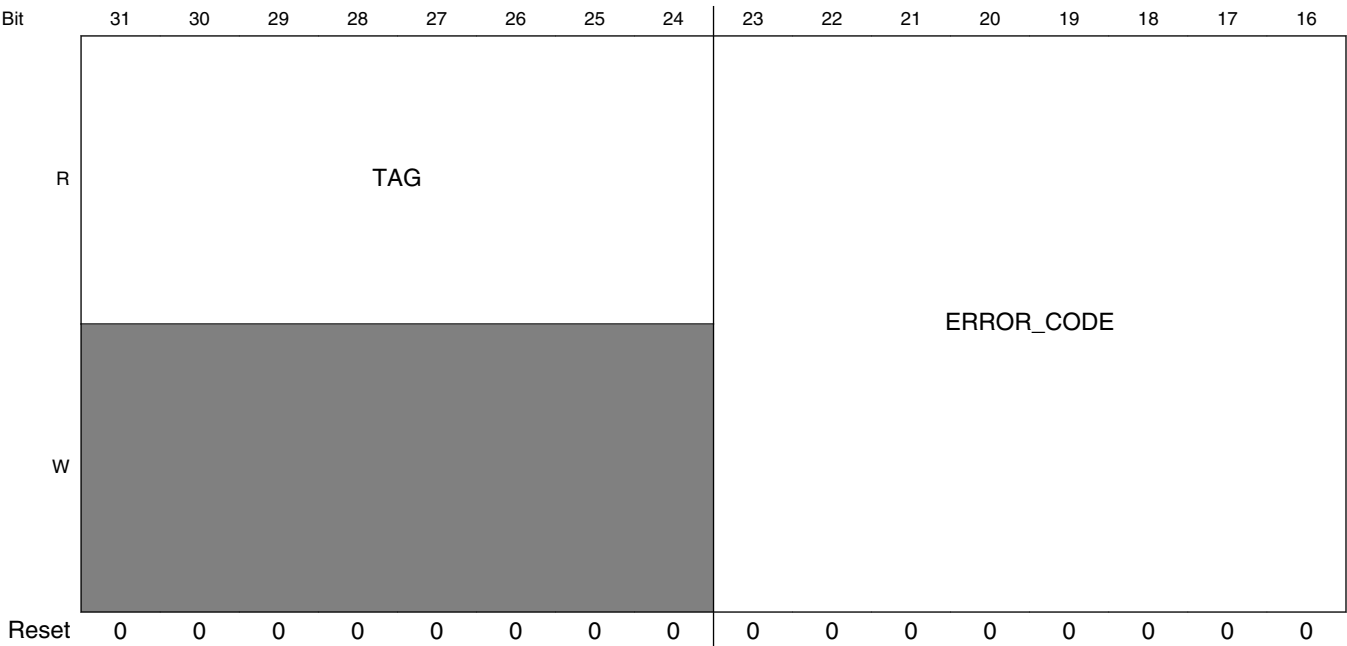
CH2STAT\_SET: 0x1A4

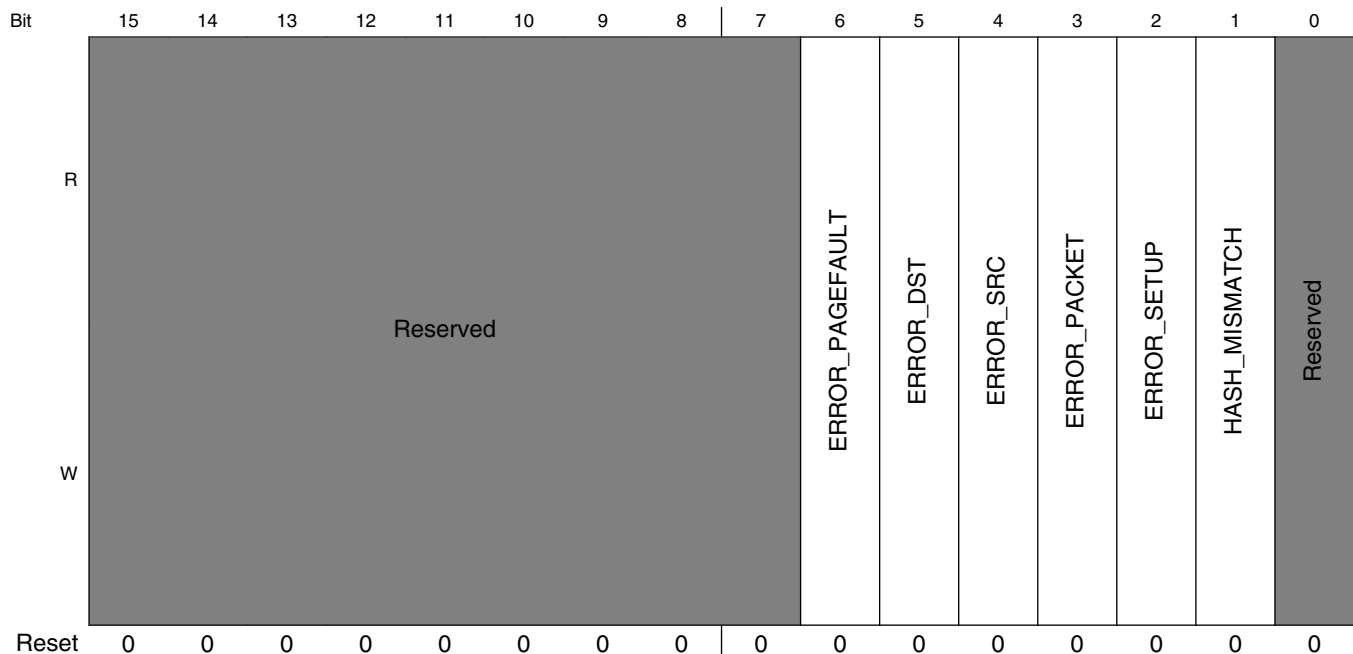
CH2STAT\_CLR: 0x1A8

CH2STAT\_TOG: 0x1AC

The interrupt status register is updated at the end of each work packet. If the interrupt bit is set in the command packet's command field, an interrupt will be generated once the packet has completed. In addition, the tag value from the command is stored in the TAG field so that software can identify which command structure was the last to complete. If an error occurs, the ERROR bit is set and processing of the command chain is halted.

Address: 20F\_C000h base + 1A0h offset = 20F\_C1A0h





DCP\_CH2STAT field descriptions

Field	Description
31–24 TAG	Indicates the tag from the last completed packet in the command structure
23–16 ERROR_CODE	Indicates additional error codes for some error conditions. 0x01 <b>NEXT_CHAIN_IS_0</b> — Error signalled because the next pointer is 0x00000000 0x02 <b>NO_CHAIN</b> — Error signalled because the semaphore is nonzero and neither chain bit is set 0x03 <b>CONTEXT_ERROR</b> — Error signalled because an error was reported reading/writing the context buffer 0x04 <b>PAYLOAD_ERROR</b> — Error signalled because an error was reported reading/writing the payload 0x05 <b>INVALID_MODE</b> — Error signalled because the control packet specifies an invalid mode select (for instance, blit + hash)
15–7 -	This field is reserved. Reserved, always set to zero.
6 ERROR_PAGEFAULT	This bit indicates a page fault occurred while converting a virtual address to a physical address.. When an error is detected, the channel's processing will stop until the error handled by software.
5 ERROR_DST	This bit indicates a bus error occurred when storing to the destination buffer. When an error is detected, the channel's processing will stop until the error handled by software.
4 ERROR_SRC	This bit indicates a bus error occurred when reading from the source buffer. When an error is detected, the channel's processing will stop until the error handled by software.
3 ERROR_PACKET	This bit indicates that a bus error occurred when reading the packet or payload or when writing status back to the packet payload. When an error is detected, the channel's processing will stop until the error is handled by software.
2 ERROR_SETUP	This bit indicates that the hardware detected an invalid programming configuration such as a buffer length that is not a multiple of the natural data size for the operation. When an error is detected, the channel's processing will stop until the error is handled by software.

Table continues on the next page...

**DCP\_CH2STAT field descriptions (continued)**

Field	Description
1 HASH_ MISMATCH	The bit indicates that a hashing check operation mismatched for control packets that enable the HASH_CHECK bit. When an error is detected, the channel's processing will stop until the error is handled by software.
0 RSVD_ COMPLETE	This field is reserved. This bit will always read 0 in the status register, but will be set to 1 in the packet status field after processing of the packet has completed. This was done so that software can verify that each packet completed properly in a chain of commands for cases when an interrupt is issued only for the last item in a packet. The completion bit for the channel is effectively the channel interrupt status bit.

**18.4.27 DCP Channel 2 Options Register (DCP\_CH2OPTS)**

The DCP Channel 2 Options Status register contains optional control information that may be used to further tune the behavior of the channel.

CH2OPTS: 0x1B0

CH2OPTS\_SET: 0x1B4

CH2OPTS\_CLR: 0x1B8

CH2OPTS\_TOG: 0x1BC

The options register can be used to control optional features of the channels.

Address: 20F\_C000h base + 1B0h offset = 20F\_C1B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DCP\_CH2OPTS field descriptions**

Field	Description
31–16 RSVD	This field is reserved. Reserved, always set to zero.
15–0 RECOVERY_ TIMER	This field indicates the recovery time for the channel. After each operation, the recover timer for the channel is initialized with this value and then decremented until the timer reaches zero. The channel will not initiate operation on the next packet in the chain until the recovery time has been satisfied. The timebase for the recovery timer is 16 HCLK clock cycles, providing a range of 0ns to 8.3ms at 133 MHz operation.



### 18.4.28 DCP Channel 3 Command Pointer Address Register (DCP\_CH3CMDPTR)

The DCP channel 3 current command address register points to the multiword descriptor that is to be executed (or currently being executed). The channel may be activated by writing the command pointer address to a valid descriptor in memory and then updating the semaphore to a non-zero value. After the engine completes processing of a descriptor, the "next\_ptr" field from the descriptor is moved into this register to enable processing of the next descriptor. All channels with a non-zero semaphore value will arbitrate for access to the engine for the subsequent operation.

DCP Channel 3 is controlled by a variable sized command structure. This register points to the command structure to be executed.

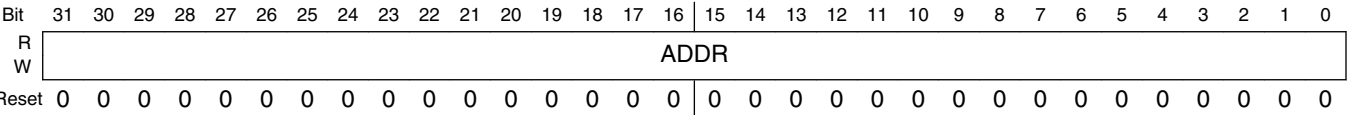
#### EXAMPLE

```

DCP_CHn_CMDPTR_WR(3, v); // Write channel 3 command pointer
pCurptr = (DCP_chn_cmdptr_t *) DCP_CHn_CMDPTR_RD(3); // Read current command
pointer

```

Address: 20F\_C000h base + 1C0h offset = 20F\_C1C0h



DCP\_CH3CMDPTR field descriptions

Field	Description
31–0 ADDR	Pointer to descriptor structure to be processed for channel 3.

## 18.4.29 DCP Channel 3 Semaphore Register (DCP\_CH3SEMA)

The DCP Channel 3 semaphore register is used to synchronize the ARM platform instruction stream and the DMA chain processing state. After a command chain has been generated in memory, software should write the address of the first command descriptor to the CMDPTR register and then write a non-zero value to the semaphore register to indicate that the channel is active. Each command packet has a chaining bit which indicates that another descriptor should be loaded into the channel upon completion of the current descriptor. If the chaining bit is not set, the next address will not be loaded into the CMDPTR register. Each packet also contains a "decrement semaphore" bit, which indicates that the counting semaphore should be decremented after the operation. A channel is considered active when the semaphore is a non-zero value. When programming a series operations, software must properly program the semaphore values in conjunction with the "decrement\_semaphore" bits in the control packets to ensure that the proper number of descriptors are activated. A semaphore may be cleared by software by writing 0xFF to the DCP\_CHnSEMA\_CLR register. The logic will also clear the semaphore if an error has occurred.

Each DCP channel has an 8 bit counting semaphore that is used to synchronize between the program stream and the DCP chain processing. DCP processing continues until the engine attempts to decrement a semaphore that has already reached a value of zero. When the attempt is made, the DCP channel is stalled until software increments the semaphore count.

Address: 20F\_C000h base + 1D0h offset = 20F\_C1D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								VALUE								Reserved								INCREMENT							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DCP\_CH3SEMA field descriptions**

Field	Description
31–24 -	This field is reserved. Reserved, always set to zero.
23–16 VALUE	This read-only field shows the current (instantaneous) value of the semaphore counter.
15–8 -	This field is reserved. Reserved, always set to zero.
7–0 INCREMENT	The value written to this field is added to the semaphore count in an atomic way such that simultaneous software adds and DCP hardware subtracts happening on the same clock are protected. This bit field reads back a value of 0x00. Writing a value of 0x02 increments the semaphore count by two, unless the DCP channel decrements the count on the same clock, then the count is incremented by a net one. The semaphore may be cleared by writing 0xFF to the DCP_CHnSEMA_CLR register.

### 18.4.30 DCP Channel 3 Status Register (DCP\_CH3STAT)

The DCP Channel 3 Interrupt Status register contains the interrupt status bit and the tag of the last completed operation from the command chain. If an error occurs during processing, the ERROR bit is set and an interrupt is generated.

DCP\_CH3STAT: 0x1E0

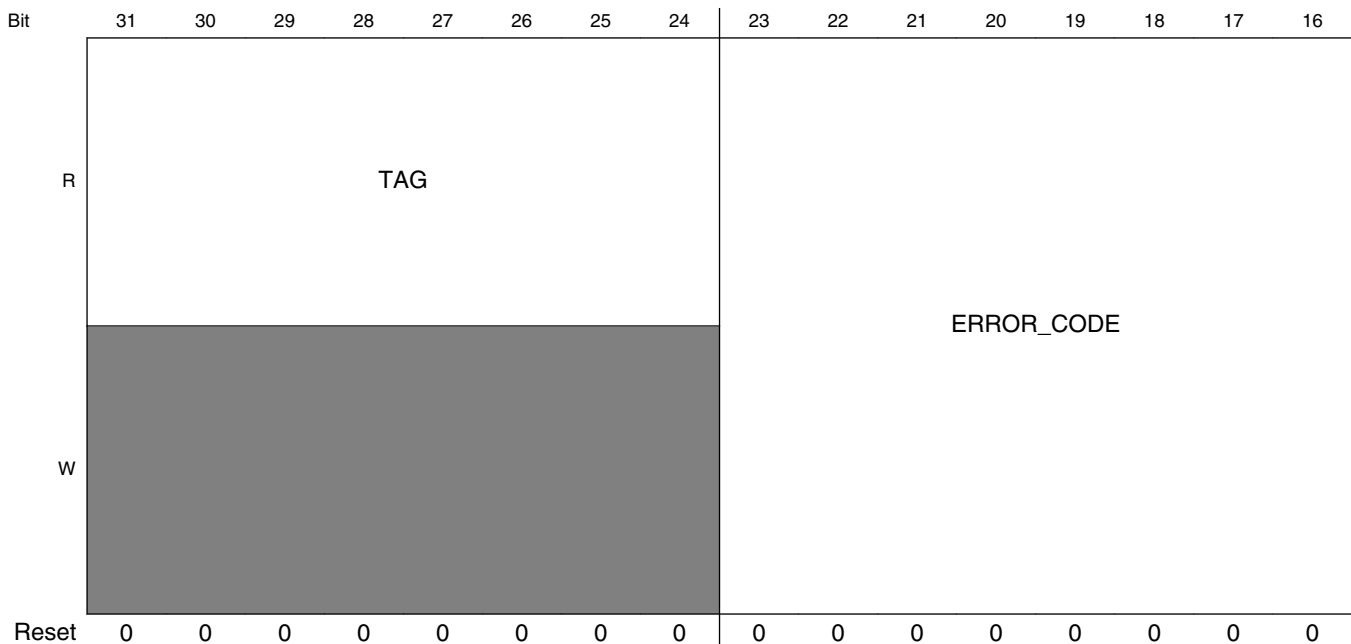
CH3STAT\_SET: 0x1E4

CH3STAT\_CLR: 0x1E8

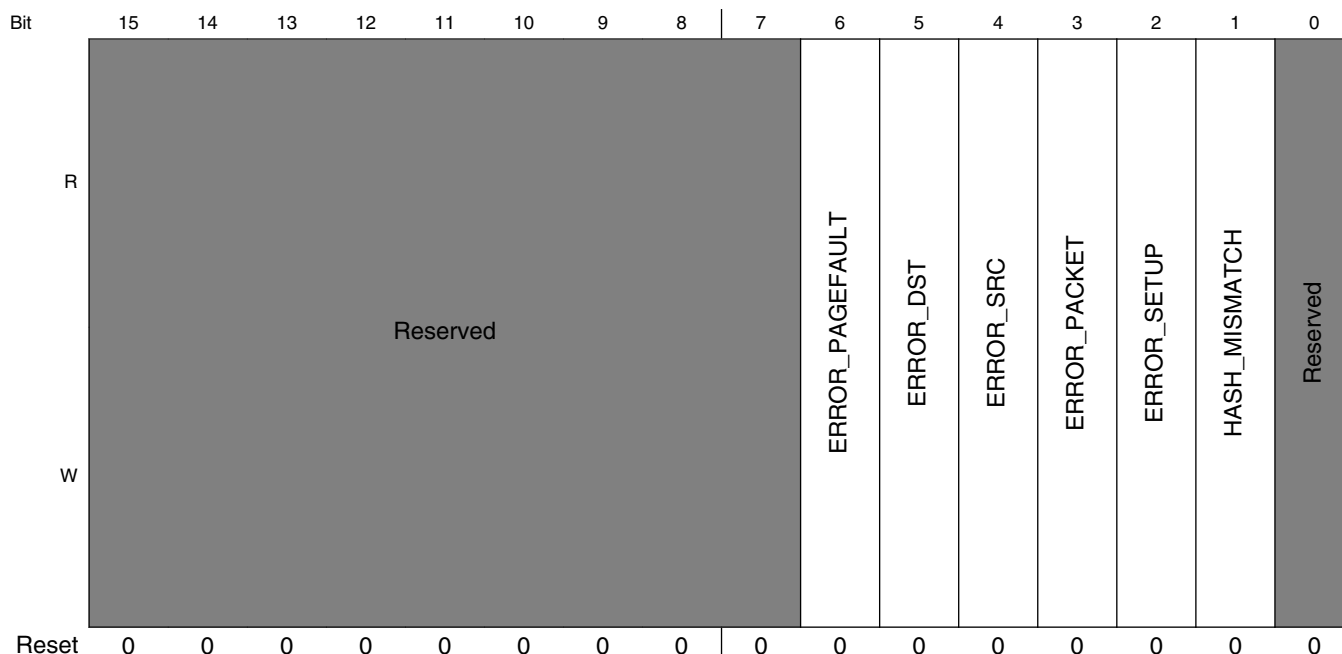
CH3STAT\_TOG: 0x1EC

The interrupt status register is updated at the end of each work packet. If the interrupt bit is set in the command packet's command field, an interrupt will be generated once the packet has completed. In addition, the tag value from the command is stored in the TAG field so that software can identify which command structure was the last to complete. If an error occurs, the ERROR bit is set and processing of the command chain is halted.

Address: 20F\_C000h base + 1E0h offset = 20F\_C1E0h



## DCP Memory Map/Register Definition



**DCP\_CH3STAT field descriptions**

Field	Description
31–24 TAG	Indicates the tag from the last completed packet in the command structure
23–16 ERROR_CODE	Indicates additional error codes for some error conditions. 0x01 <b>NEXT_CHAIN_IS_0</b> — Error signalled because the next pointer is 0x00000000 0x02 <b>NO_CHAIN</b> — Error signalled because the semaphore is nonzero and neither chain bit is set 0x03 <b>CONTEXT_ERROR</b> — Error signalled because an error was reported reading/writing the context buffer 0x04 <b>PAYLOAD_ERROR</b> — Error signalled because an error was reported reading/writing the payload 0x05 <b>INVALID_MODE</b> — Error signalled because the control packet specifies an invalid mode select (for instance, blit + hash)
15–7 -	This field is reserved. Reserved, always set to zero.
6 ERROR_PAGEFAULT	This bit indicates a page fault occurred while converting a virtual address to a physical address.. When an error is detected, the channel's processing will stop until the error handled by software.
5 ERROR_DST	This bit indicates a bus error occurred when storing to the destination buffer. When an error is detected, the channel's processing will stop until the error handled by software.
4 ERROR_SRC	This bit indicates a bus error occurred when reading from the source buffer. When an error is detected, the channel's processing will stop until the error handled by software.
3 ERROR_PACKET	This bit indicates that a bus error occurred when reading the packet or payload or when writing status back to the packet payload. When an error is detected, the channel's processing will stop until the error is handled by software.
2 ERROR_SETUP	This bit indicates that the hardware detected an invalid programming configuration such as a buffer length that is not a multiple of the natural data size for the operation. When an error is detected, the channel's processing will stop until the error is handled by software.

*Table continues on the next page...*

**DCP\_CH3STAT field descriptions (continued)**

Field	Description
1 HASH_ MISMATCH	The bit indicates that a hashing check operation mismatched for control packets that enable the HASH_CHECK bit. When an error is detected, the channel's processing will stop until the error is handled by software.
0 RSVD_ COMPLETE	This field is reserved. This bit will always read 0 in the status register, but will be set to 1 in the packet status field after processing of the packet has completed. This was done so that software can verify that each packet completed properly in a chain of commands for cases when an interrupt is issued only for the last item in a packet. The completion bit for the channel is effectively the channel interrupt status bit.

**18.4.31 DCP Channel 3 Options Register (DCP\_CH3OPTS)**

The DCP Channel 3 Options Status register contains optional control information that may be used to further tune the behavior of the channel.

DCP\_CH3OPTS: 0x1F0

CH3OPTS\_SET: 0x1F4

CH3OPTS\_CLR: 0x1F8

CH3OPTS\_TOG: 0x1FC

The options register can be used to control optional features of the channels.

Address: 20F\_C000h base + 1F0h offset = 20F\_C1F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																RECOVERY_TIMER															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**DCP\_CH3OPTS field descriptions**

Field	Description
31–16 RSVD	This field is reserved. Reserved, always set to zero.
15–0 RECOVERY_ TIMER	This field indicates the recovery time for the channel. After each operation, the recover timer for the channel is initialized with this value and then decremented until the timer reaches zero. The channel will not initiate operation on the next packet in the chain until the recovery time has been satisfied. The timebase for the recovery timer is 16 HCLK clock cycles, providing a range of 0ns to 8.3ms at 133 MHz operation.

**18.4.32 DCP Debug Select Register (DCP\_DBGSELECT)**

This register selects a debug register to view.

This register selects debug information to return in the debug data register.

Address: 20F\_C000h base + 400h offset = 20F\_C400h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																INDEX															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### DCP\_DBGSELECT field descriptions

Field	Description
31–8 RSVD	This field is reserved. Reserved, always set to zero.
7–0 INDEX	Selects a value to read via the debug data register.  0x01 <b>CONTROL</b> — 0x10 <b>OTPKEY0</b> — 0x11 <b>OTPKEY1</b> — 0x12 <b>OTPKEY2</b> — 0x13 <b>OTPKEY3</b> —

### 18.4.33 DCP Debug Data Register (DCP\_DBGDATA)

Reading this register returns the debug data value from the selected index.

This register returns the debug data from the selected debug index source.

Address: 20F\_C000h base + 410h offset = 20F\_C410h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### DCP\_DBGDATA field descriptions

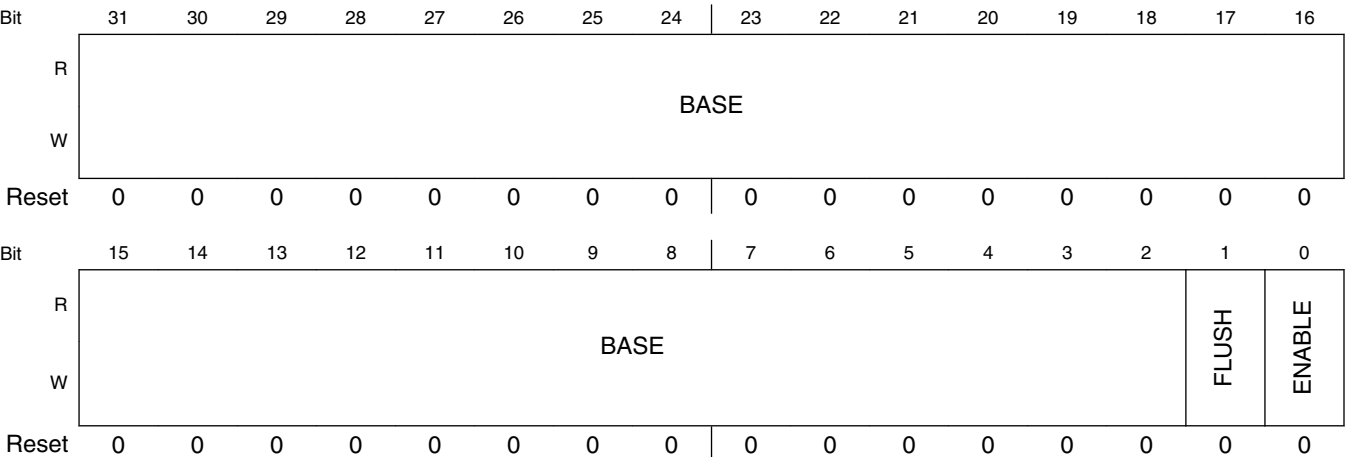
Field	Description
31–0 DATA	Debug Data

### 18.4.34 DCP Page Table Register (DCP\_PAGETABLE)

The DCP Page Table register controls the virtual memory functionality of the DCP. It provides a base address for the page table as well as an enable/disable bit and the ability to flush the cached page table entries.

This register returns the debug data from the selected debug index source.

Address: 20F\_C000h base + 420h offset = 20F\_C420h



DCP\_PAGETABLE field descriptions

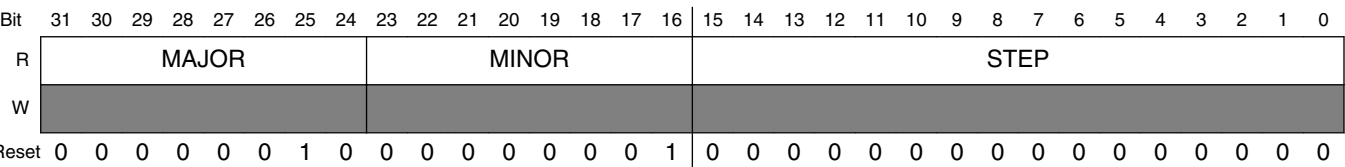
Field	Description
31–2 BASE	Page Table Base Address. The page table must be word aligned and the pointer should reference a page table in the standard ARM format.
1 FLUSH	Page Table Flush control. To flush the TLB, write this bit to a 1 then back to a 0.
0 ENABLE	Page Table Enable control. Virtual addressing will only be used when this bit is set to a 1. Disabling the page table will not flush any cached entries, so software should write the FLUSH high and enable LOW when updating page tables.

18.4.35 DCP Version Register (DCP\_VERSION)

Read-only register indicating implemented version of the DCP.

This register returns the debug data from the selected debug index source.

Address: 20F\_C000h base + 430h offset = 20F\_C430h



DCP\_VERSION field descriptions

Field	Description
31–24 MAJOR	Fixed read-only value reflecting the MAJOR version of the design implementation.

Table continues on the next page...

**DCP\_VERSION field descriptions (continued)**

Field	Description
23–16 MINOR	Fixed read-only value reflecting the MINOR version of the design implementation.
15–0 STEP	Fixed read-only value reflecting the stepping of version of the design implementation.



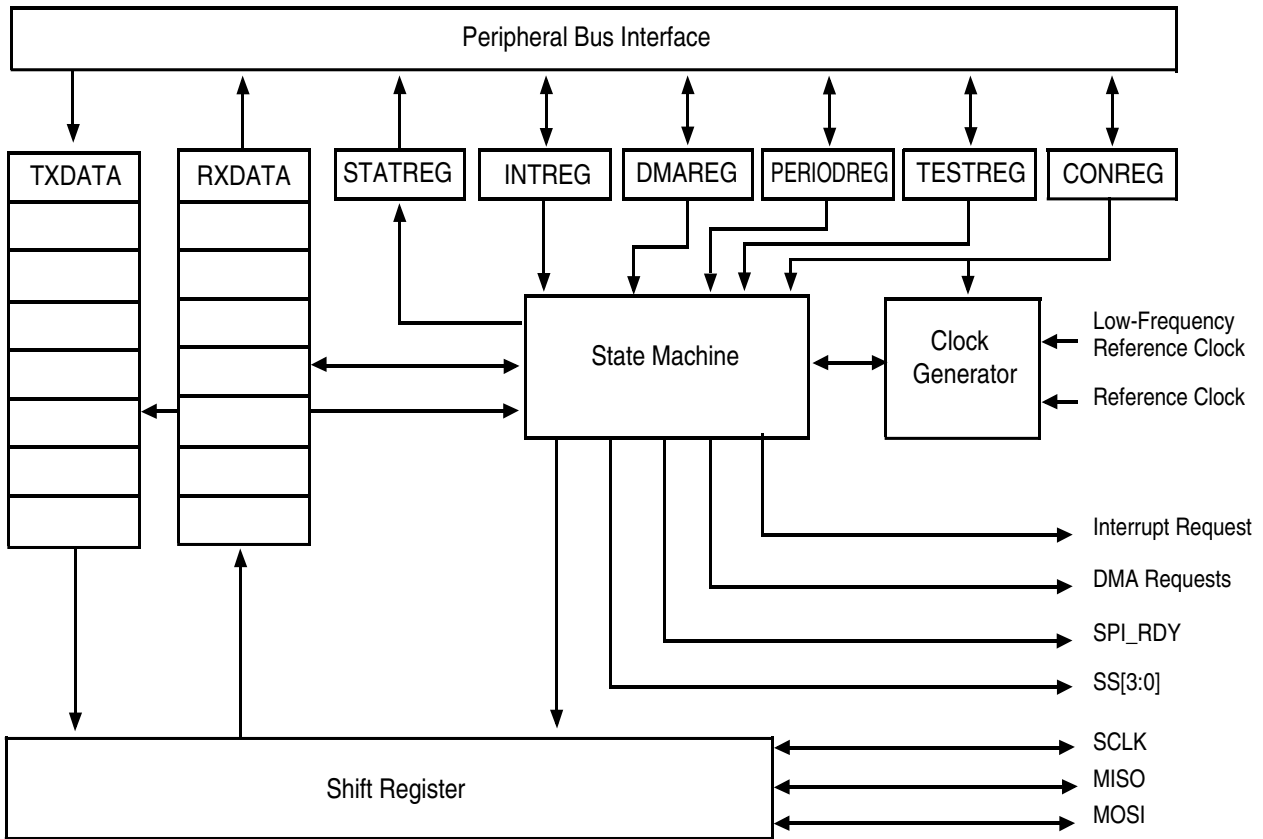
## Chapter 19

# Enhanced Configurable SPI (ECSPI)

### 19.1 Overview

The Enhanced Configurable Serial Peripheral Interface (ECSPI) is a full-duplex, synchronous, four-wire serial communication block.

The ECSPI contains a 64 x 32 receive buffer (RXFIFO) and a 64 x 32 transmit buffer (TXFIFO). With data FIFOs, the ECSPI allows rapid data communication with fewer software interrupts. The figure below shows a block diagram of the ECSPI.



**Figure 19-1. ECSPI Block Diagram**

### 19.1.1 Features

Key features of the ECSPI include:

- Full-duplex synchronous serial interface
- Master/Slave configurable
- Four Chip Select (SS) signals to support multiple peripherals
- Transfer continuation function allows unlimited length data transfers
- 32-bit wide by 64-entry FIFO for both transmit and receive data
- Polarity and phase of the Chip Select (SS) and SPI Clock (SCLK) are configurable
- Direct Memory Access (DMA) support
- Max operation frequency up to the reference clock frequency.

## 19.1.2 Modes and Operations

The ECSPI supports the modes described in the indicated sections:

- [Master Mode](#)
- [Slave Mode](#)
- [Low Power Modes](#)

As described in [Operations](#), the ECSPI supports the operations described in the indicated sections:

- [Typical Master Mode](#)
  - [Master Mode with SPI\\_RDY](#)
  - [Master Mode with Wait States](#)
  - [Master Mode with SS\\_CTL\[3:0\] Control](#)
  - [Master Mode with Phase Control](#)
- [Typical Slave Mode](#)

## 19.2 External Signals

The following table describes the external signals of ECSPI:

**Table 19-1. ECSPI1 External Signals**

Signal	Description	Pad	Mode	Direction
ECSPI1_MISO (MISO)	Master data in; slave data out	ECSPI1_MISO	ALT0	IO
		LCD_DAT1	ALT1	
ECSPI1_MOSI (MOSI)	Master data out; slave data in	ECSPI1_MOSI	ALT0	IO
		LCD_DAT0	ALT1	
ECSPI1_RDY (RDY)	SPI data ready signal	I2C2_SCL	ALT6	I
		LCD_DAT7	ALT1	
ECSPI1_SCLK (SCLK)	SPI clock signal	ECSPI1_SCLK	ALT0	IO
		LCD_DAT3	ALT1	
ECSPI1_SS0 (SS0)	Chip select signal	ECSPI1_SS0	ALT0	IO
		LCD_DAT2	ALT1	
ECSPI1_SS1 (SS1)	Chip select signal	I2C1_SCL	ALT6	IO
		LCD_DAT4	ALT1	
ECSPI1_SS2 (SS2)	Chip select signal	I2C1_SDA	ALT6	IO
		LCD_DAT5	ALT1	
ECSPI1_SS3 (SS3)	Chip select signal	ECSPI2_SS0	ALT1	IO
		LCD_DAT6	ALT1	

**Table 19-2. ECSPi2 External Signals**

Signal	Description	Pad	Mode	Direction
ECSPi2_MISO (MISO)	Master data in; slave data out	ECSPi2_MISO	ALT0	IO
		EPDC_SDLE	ALT1	
		LCD_DAT10	ALT4	
ECSPi2_MOSI (MOSI)	Master data out; slave data in	ECSPi2_MOSI	ALT0	IO
		EPDC_SDCLK	ALT1	
		LCD_DAT9	ALT4	
ECSPi2_RDY (RDY)	SPI data ready signal	EPDC_GDRL	ALT1	I
ECSPi2_SCLK (SCLK)	SPI clock signal	ECSPi2_SCLK	ALT0	IO
		EPDC_SDSHR	ALT1	
		LCD_DAT8	ALT4	
ECSPi2_SS0 (SS0)	Chip select signal	ECSPi2_SS0	ALT0	IO
		EPDC_SDOE	ALT1	
ECSPi2_SS1 (SS1)	Chip select signal	EPDC_SDCE0	ALT1	IO
		LCD_DAT11	ALT4	
ECSPi2_SS2 (SS2)	Chip select signal	EPDC_GDCLK	ALT1	IO
ECSPi2_SS3 (SS3)	Chip select signal	EPDC_GDOE	ALT1	IO

**Table 19-3. ECSPi3 External Signals**

Signal	Description	Pad	Mode	Direction
ECSPi3_MISO (MISO)	Master data in; slave data out	AUD_TXC	ALT1	IO
		EPDC_D9	ALT1	
		SD2_DAT1	ALT2	
ECSPi3_MOSI (MOSI)	Master data out; slave data in	AUD_RXD	ALT1	IO
		EPDC_D8	ALT1	
		SD2_DAT0	ALT2	
ECSPi3_RDY (RDY)	SPI data ready signal	AUD_MCLK	ALT2	I
		EPDC_D15	ALT6	
ECSPi3_SCLK (SCLK)	SPI clock signal	AUD_TXD	ALT1	IO
		EPDC_D11	ALT1	
		SD2_CLK	ALT2	
ECSPi3_SS0 (SS0)	Chip select signal	AUD_RXFS	ALT6	IO
		EPDC_D10	ALT1	
		SD2_CMD	ALT2	
ECSPi3_SS1 (SS1)	Chip select signal	AUD_RXC	ALT6	IO
		EPDC_D12	ALT6	
ECSPi3_SS2 (SS2)	Chip select signal	EPDC_D13	ALT6	IO
		I2C1_SCL	ALT2	
ECSPi3_SS3 (SS3)	Chip select signal	EPDC_D14	ALT6	IO
		I2C1_SDA	ALT2	

**Table 19-4. ECSPI4 External Signals**

Signal	Description	Pad	Mode	Direction
ECSPI4_MISO (MISO)	Master data in; slave data out	EPDC_D1	ALT1	IO
		FEC_CRS_DV	ALT3	
		KEY_ROW1	ALT1	
ECSPI4_MOSI (MOSI)	Master data out; slave data in	EPDC_D0	ALT1	IO
		FEC_RX_ER	ALT3	
		KEY_COL1	ALT1	
ECSPI4_RDY (RDY)	SPI data ready signal	EPDC_D7	ALT1	I
ECSPI4_SCLK (SCLK)	SPI clock signal	EPDC_D3	ALT1	IO
		FEC_TX_CLK	ALT3	
		KEY_ROW2	ALT1	
ECSPI4_SS0 (SS0)	Chip select signal	EPDC_D2	ALT1	IO
		FEC_MDIO	ALT3	
		KEY_COL2	ALT1	
ECSPI4_SS1 (SS1)	Chip select signal	EPDC_D4	ALT1	IO
		FEC_RXD1	ALT3	
ECSPI4_SS2 (SS2)	Chip select signal	EPDC_D5	ALT1	IO
		FEC_TXD0	ALT3	
ECSPI4_SS3 (SS3)	Chip select signal	EPDC_D6	ALT1	IO

Figure 19-2 shows the ECSPI in master mode connected to four external devices in a one-way communication link.

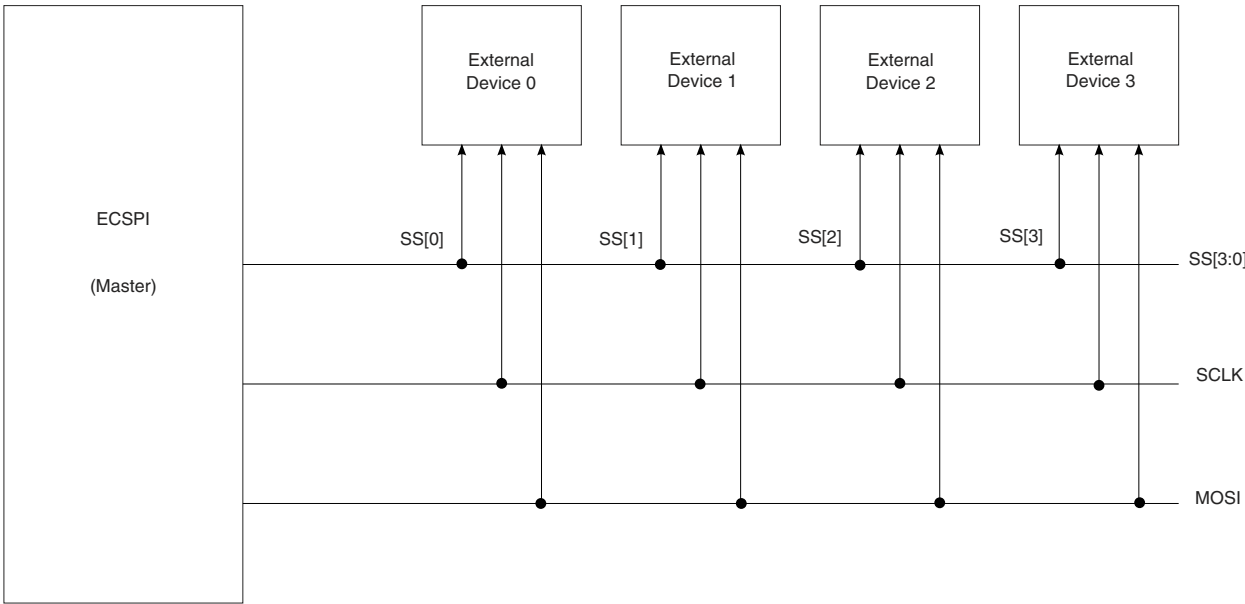


Figure 19-2. Example Connection Diagram

19.3 Clocks

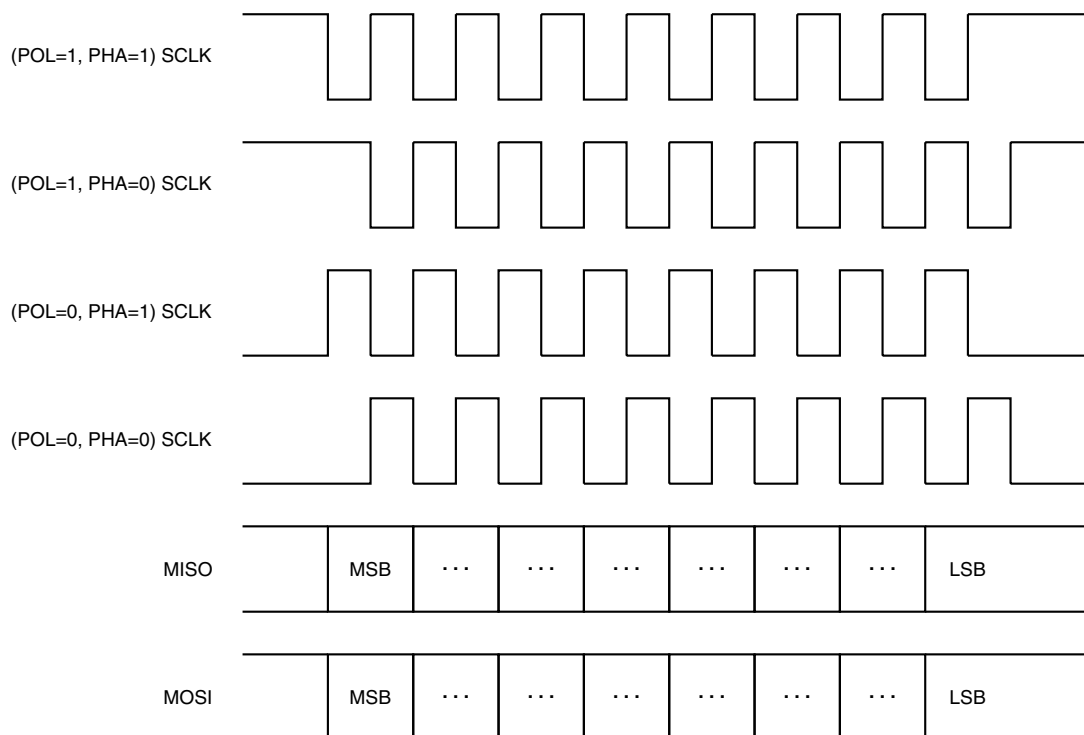
The following table describes the clock sources for eCSPI. Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

Table 19-5. eCSPI Clocks

Clock name	Clock Root	Description
ipg_clk	ipg_clk_root	Peripheral clock
ipg_clk_32k	ckil_sync_clk_root	Low-frequency reference clock (32kHz)
ipg_clk_per	ecspi_clk_root	eCSPI module clock
ipg_clk_s	ipg_clk_root	Peripheral access clock

## 19.4 Functional Description

This section provides a complete functional description of the ECSPI. The figure found here shows the relationship of SCLK and data lines while ECSPI has been configured with different POL and PHA settings.



**Figure 19-3. ECSPI SCLK, MISO, and MOSI Relationship**

### 19.4.1 Master Mode

When the ECSPI is configured as a master, it uses a serial link to transfer data between the ECSPI and an external slave device.

One of the Chip Select (SS) signals and the clock signal (SCLK) are used to transfer data between two devices. If the external device is a transmit-only device, the ECSPI master's output port can be ignored and used for other purposes. In order to use the internal TXFIFO and RXFIFO, two auxiliary output signals, Chip Select (SS) and SPI\_RDY, are used for data transfer rate control. Software can also configure the sample period control register to a fixed data transfer rate.

## 19.4.2 Slave Mode

When the ECSPI is configured as a slave, software can configure the ECSPI Control register to match the external SPI master's timing.

In this configuration, Chip Select ( $\overline{SS}$ ) becomes an input signal, and is used to control data transfers through the Shift register, as well as to load/store the data FIFO.

Slave mode only supports the case when SSCTL (SSB\_CTRL[x] bit) is cleared. The accurate burst length should always be specified using the BURST\_LENGTH parameter. SSCTL (SSB\_CTRL[x] bit) set to 1 is not supported in slave mode.

## 19.4.3 Low Power Modes

The ECSPI does not operate under low power mode.

It holds its operation when its clock is gated off in master mode. In slave mode, the ECSPI does not respond when its clock is gated off.

## 19.4.4 Operations

The information found here describes the ECSPI's operations.

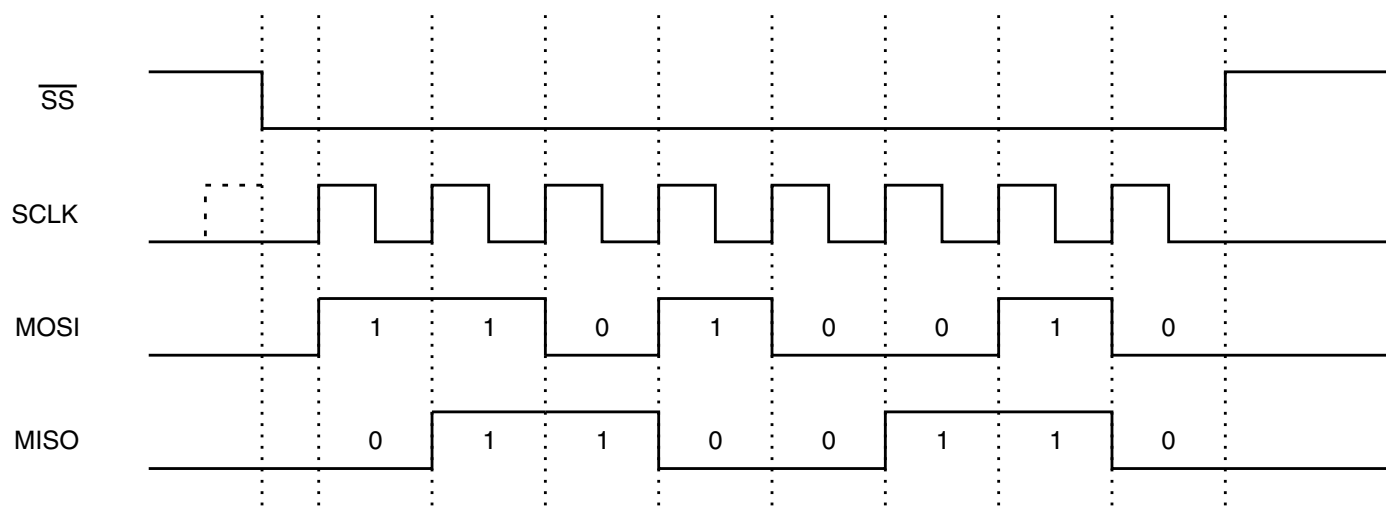
### 19.4.4.1 Typical Master Mode

The ECSPI master uses the Chip Select (SS) signal to enable an external SPI device, and uses the SCLK signal to transfer data in and out of the Shift register.

The SPI\_RDY enables fast data communication with fewer software interrupts. By programming the ECSPI\_PERIODREG register accordingly, the ECSPI can be used for a fixed data transfer rate.

When the ECSPI is in Master mode the SS, SCLK, and MOSI are output signals, and the MISO signal is an input.





**Figure 19-4. Typical SPI Burst (8-bit Transfer)**

In the above figure, the Chip Select (SS) signal enables the selected external SPI device, and the SCLK synchronizes the data transfer. The MOSI and MISO signals change on rising edge of SCLK and the MISO signal is latched on the falling edge of the SCLK. The figure above shows a data of 0xD2 is shifted out, and a data of 0x66 is shifted in.

#### 19.4.4.1.1 Master Mode with SPI\_RDY

By default, the ECSPI does not use the SPI\_RDY signal in master mode (MODE = 1).

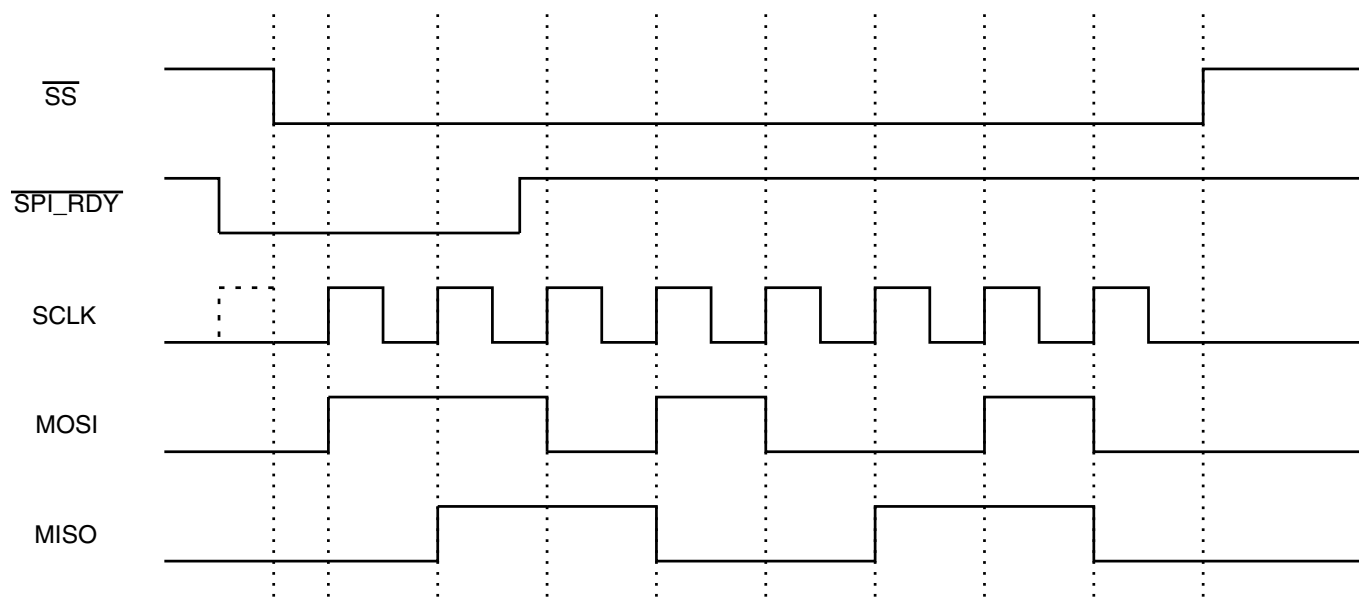
A SPI burst begins when the following events happen:

- The ECSPI is enabled, TXFIFO has data in it, and ECSPI\_CONREG[XCH] bit or the ECSPI\_CONREG[SMC] bit is set.
- When the SPI Data Ready Control (ECSPI\_CONREG[DRCTL]) bits contains either 01 or 10, the SPI\_RDY signal controls when a SPI burst starts.

A SPI burst is defined as a bus transaction that starts when the slave select is asserted and ends when the slave select is negated. The Chip Select (SS) signal will remain asserted until all the bits in a SPI burst are shifted out.

If ECSPI\_CONREG[DRCTL] is set to 01, the SPI burst can be triggered only if a falling edge of the SPI\_RDY signal has been detected.

The following figure shows the relationship between a SPI burst and the falling edge of SPI\_RDY signal.

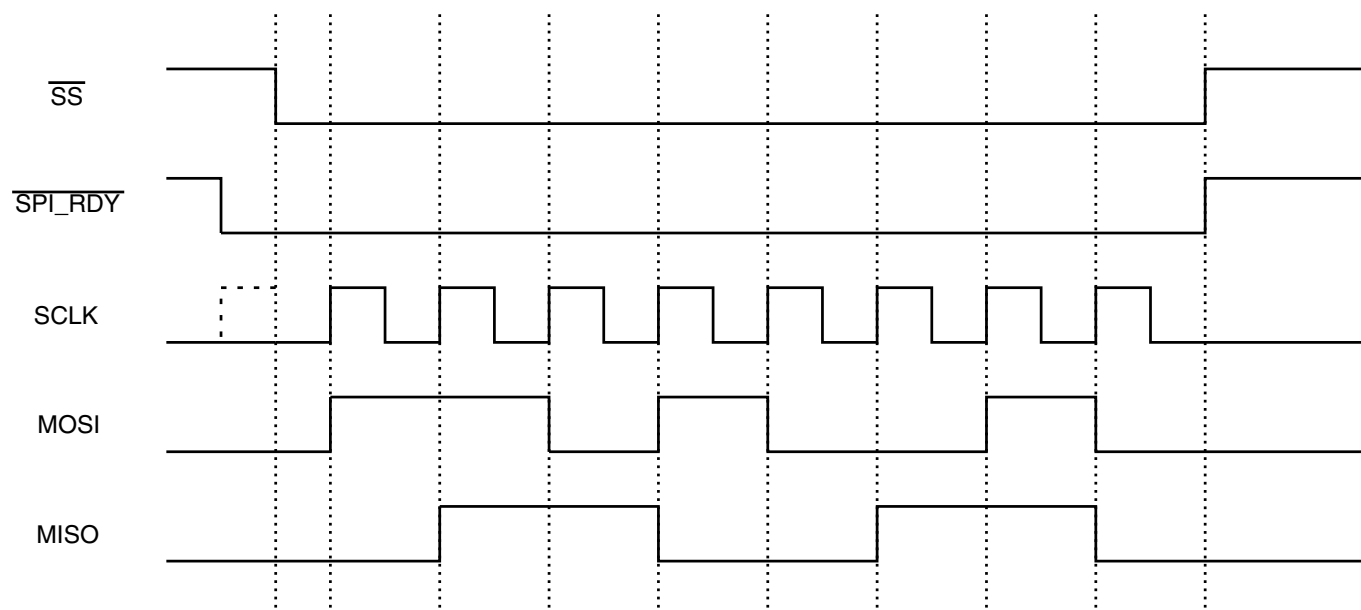


**Figure 19-5. Relationship Between a SPI Burst and SPI\_RDY: Falling-Edge Triggered**

A SPI burst does not start until the falling edge of the SPI\_RDY signal is detected. The next SPI burst starts when the next SPI\_RDY falling edge is detected, after the last burst has finished.

If SPI Data Ready Control (ECSPI\_CONREG[DRCTL]) is set to 10, the SPI burst can be triggered only if the SPI\_RDY signal is low.

The following figure shows the relationship between a SPI burst and the SPI\_RDY signal. The SPI burst does not begin until the SPI\_RDY signal goes low. The ECSPI will keep transmitting SPI burst if the SPI\_RDY signal remains low.

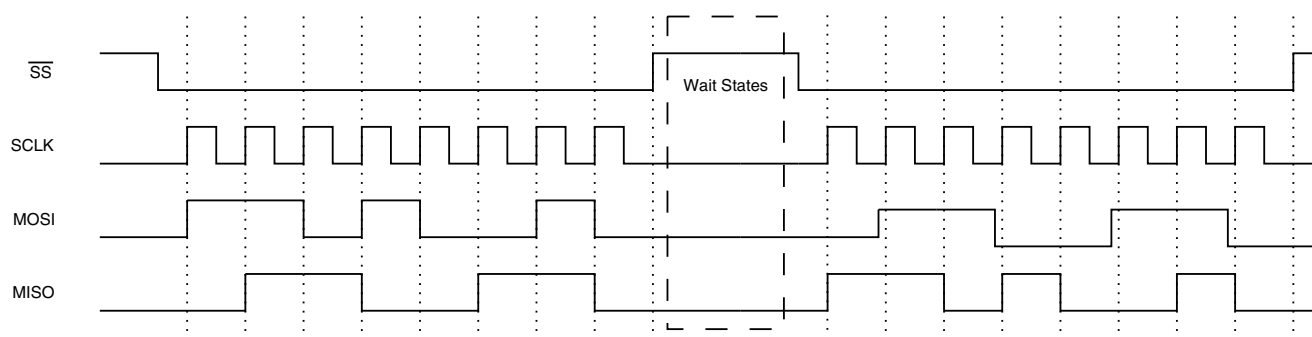


**Figure 19-6. Relationship Between a SPI Burst and SPI\_RDY: Low-Level Triggered**

#### 19.4.4.1.2 Master Mode with Wait States

Wait states can be inserted between SPI bursts. This provides a way for software to slow down the SPI burst to meet the timing requirements of a slower SPI device.

The following figure shows wait states inserted between SPI bursts.



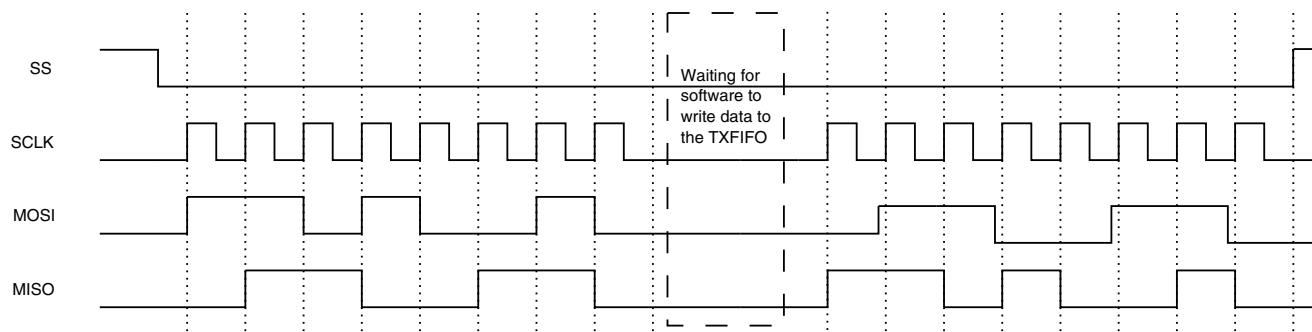
**Figure 19-7. SPI Bursts with Wait States**

In this case, the number of wait states is controlled by ECSPI\_PERIODREG[SAMPLE PERIOD] and the wait states' clock source is selected by ECSPI\_PERIODREG[CSRC].

#### 19.4.4.1.3 Master Mode with SS\_CTL[3:0] Control

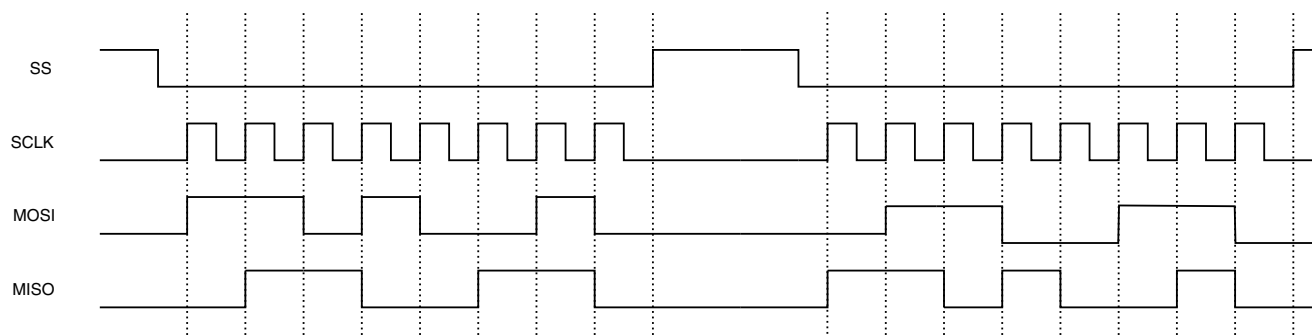
The SPI SS Control (SS\_CTL[3:0]) controls whether the current operation is single burst or multiple bursts.

When the SPI SS Wave Form Select (SS\_CTL[3:0]) is set, the current operation is multiple bursts transfer. When the SPI SS Wave Form Select (SS\_CTL[3:0]) bit is cleared, the current operation is single burst transfer. A SPI burst can contains multiple words as defined in the BURST LENGTH field of the ECSPI\_CONREG register.



**Figure 19-8. SPI Burst While SS\_CTL[3:0] is Clear**

In [Figure 19-8](#), two 8-bit bursts in the TXFIFO have been combined and transmitted in one SPI burst. The maximum length of a single SPI burst is defined in the BURST LENGTH field of the ECSPI\_CONREG control register. ([Figure 19-8](#) corresponds to a BURST LENGTH of 8.) This provides a way for transferring a longer SPI burst by writing data into TXFIFO while the ECSPI is transmitting.



**Figure 19-9. SPI Bursts While SS\_CTL[3:0] is Set**

In [Figure 19-9](#), two FIFO entries are transmitted, one entry with each SPI burst. The ECSPI will continue to transmit SPI bursts until the TXFIFO is empty. When wait states can be inserted between SPI bursts, the SS will negate between SPI bursts until the wait states finish.

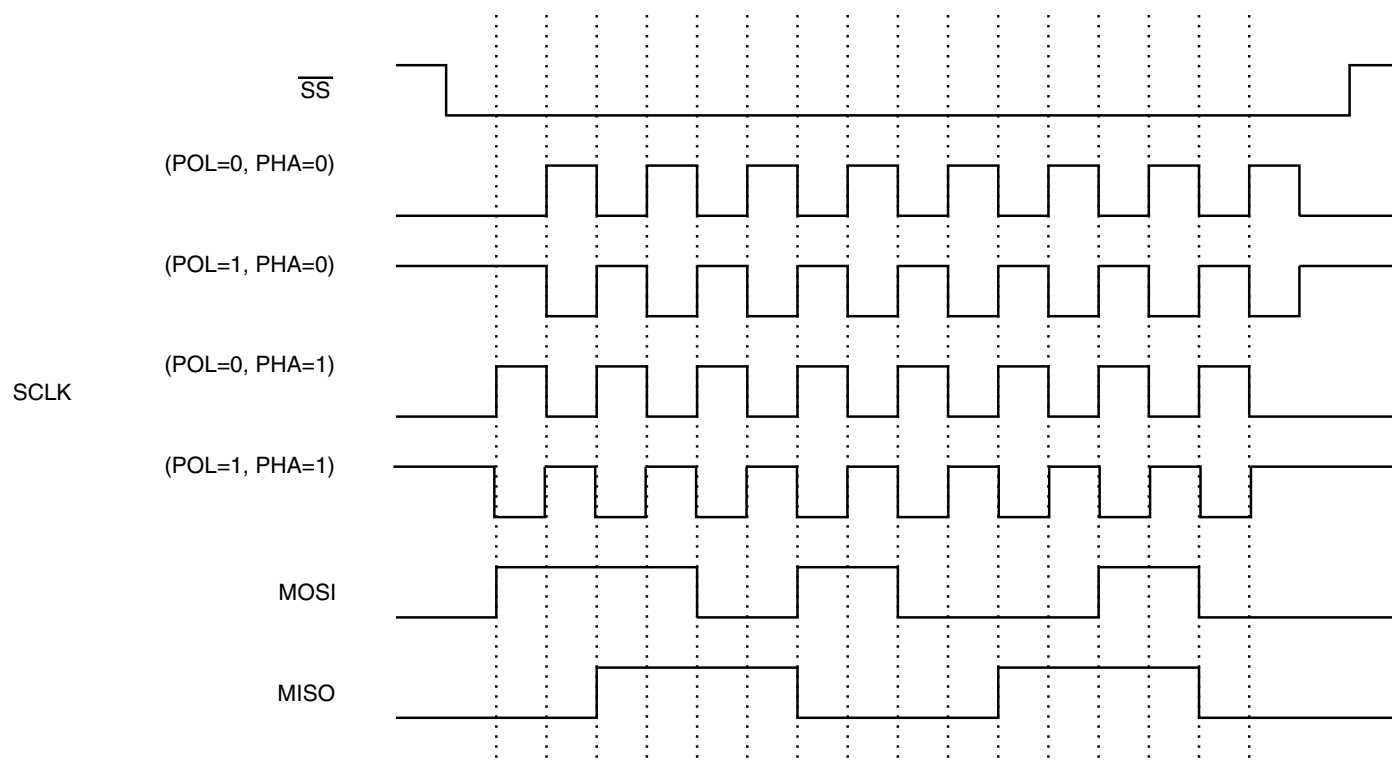
#### 19.4.4.1.4 Master Mode with Phase Control

The Phase Control (ECSPI\_CONREG[PHA]) bit controls how the transmit data shifts out and the receive data shifts in.

When the Phase control (ECSPI\_CONREG[PHA]) bit is set, the transmit data will shift out on the rising edge of SCLK, and the receive data is latched on the falling edge of SCLK. The most-significant bit is output on the first rising SCLK edge.

When ECSPI\_CONREG[PHA] is cleared, the transmit data is shifted out on the falling edge of SCLK and the receive data is latched on the rising edge of SCLK. The MSB is output when the host processor loads the transmitted data.

Inverting the SCLK polarity does not impact the edge-triggered operations because they are internal to the serial peripheral interface master. [Figure 19-10](#) shows how SPI burst works with different POL and PHA configuration.



**Figure 19-10. SPI Burst with Different POL and PHA Configurations**

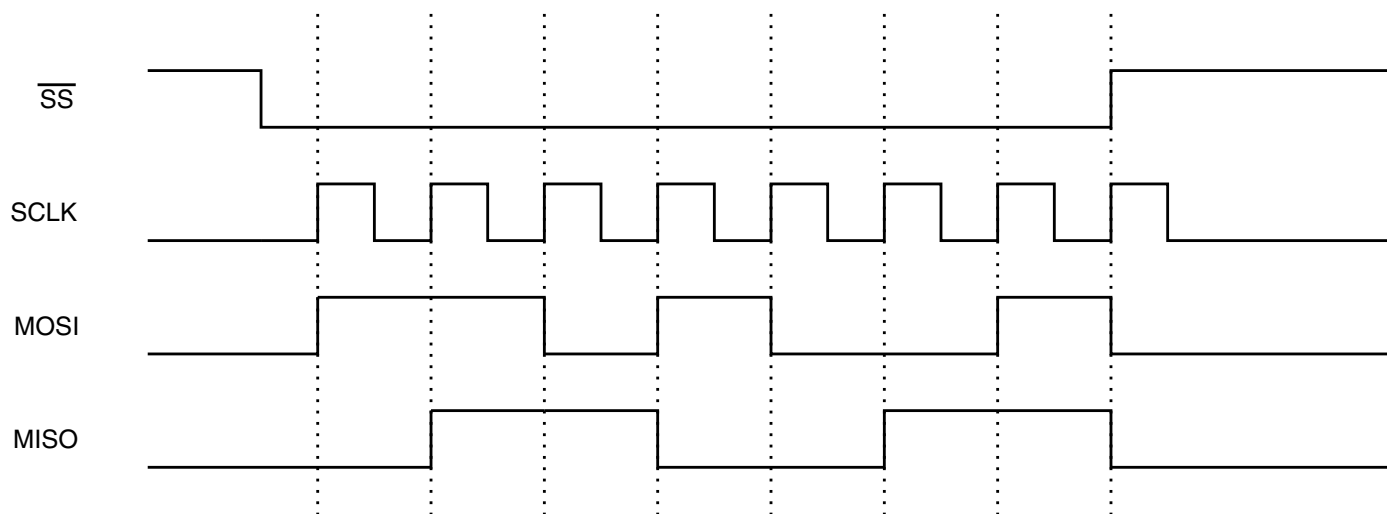
#### 19.4.4.2 Typical Slave Mode

When the ECSPI is configured as a slave (Mode = 0), software can configure the ECSPI Control register to match the external SPI master's timing. In this configuration,  $\overline{SS}$  becomes an input signal, and is used to latch data in and out of the internal data Shift registers, as well as to advance the data FIFO.

The SS, SCLK, and MOSI are inputs and MISO is output. Most of the timing diagrams are similar to the diagrams shown previously for the SPI in Master mode (Mode = 1), because the inputs come from a SPI master device.

However, the timing is different when SS is used to advance the data FIFO. When the SS\_CTL[3:0] is set while the ECSPI is configured in Slave mode, the data FIFO will advance on the rising edge of the SS signal. When the polarity is reversed (SSPOL = 1), the data FIFO will advance on the falling edge of the SS signal.

The figure below shows a SPI burst in which the data FIFO is advanced by the rising edge of the SS signal.



**Figure 19-11. Advancing the Data FIFO on the Rising Edge of  $\overline{SS}$**

In the above case, only the most significant 7 bits are loaded to the RXFIFO.

### 19.4.5 Reset

Whenever a device reset occurs, a reset is performed on the ECSPI, resetting all registers to their default values.

Software can reset the block using the CONREG[EN] bit; see [ECSPI](#).

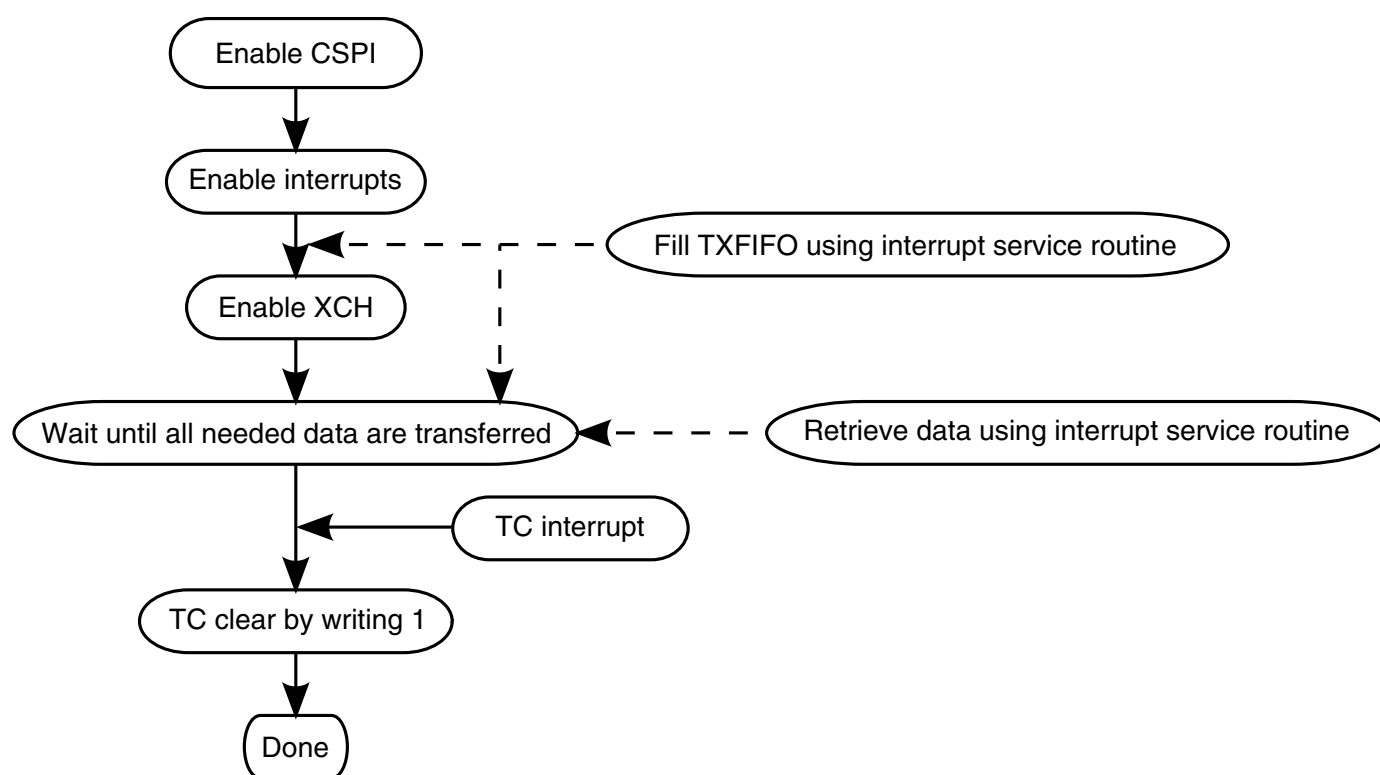
### 19.4.6 Interrupts

Interrupt control provides a way to manage the ECSPI FIFOs:

- For transmitting data, software can enable the TXFIFO empty, TXFIFO data request, and TXFIFO full interrupts to maintain the TXFIFO using an interrupt service routine.
- For receiving data, software can enable the RXFIFO ready, RXFIFO data request, and RXFIFO full interrupts to retrieve data from the RXFIFO using an interrupt service routine.

Other interrupt sources can be used to control or debug the SPI bursts:

- The transfer-completed interrupt means that there is no data left in the TXFIFO and that the data in the Shift register has been shifted out.
- The RXFIFO overflow interrupt means that the RXFIFO received more than 64 words and will not accept any other words.



**Figure 19-12. Program Sequence of SPI Burst Using Interrupt Control**

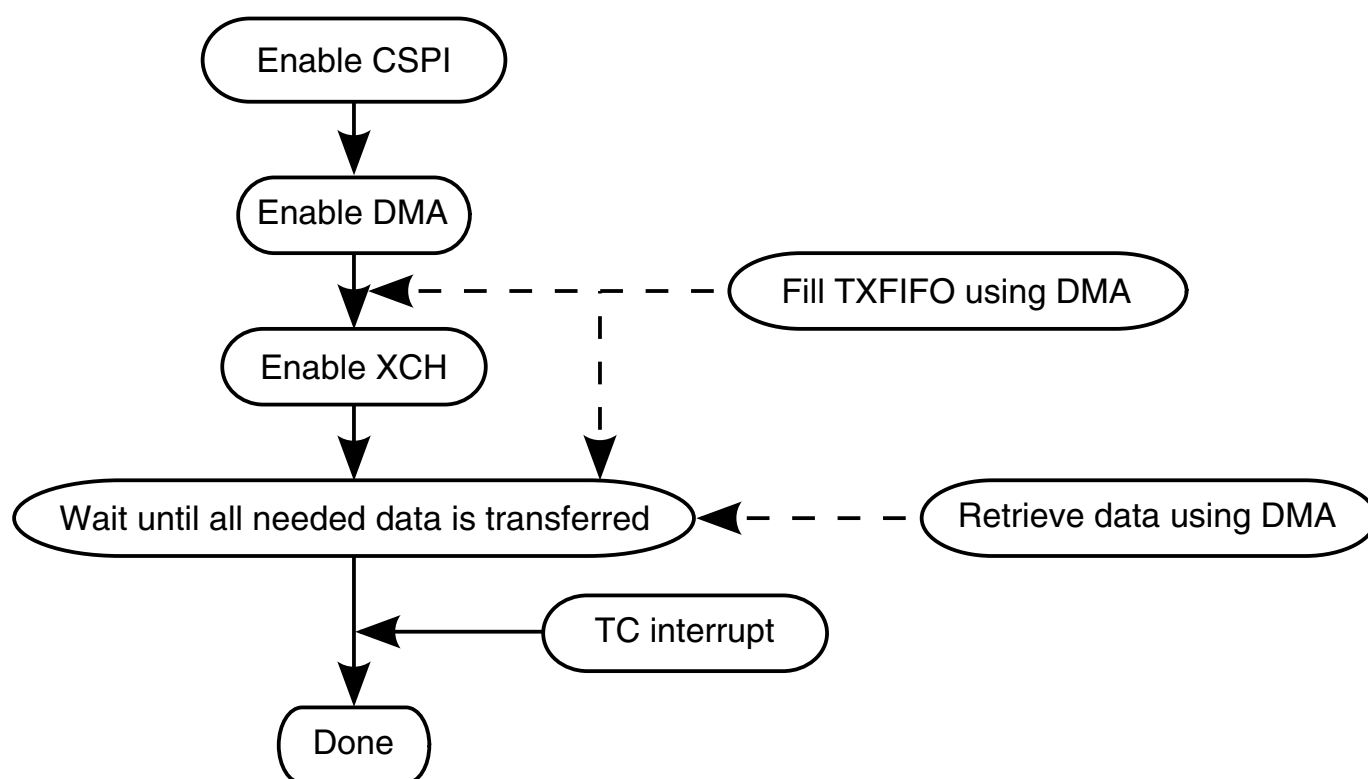
## 19.4.7 DMA

DMA control provides another method to utilize the FIFOs in the ECSPI. By using DMA request and acknowledge signals, larger amounts of data can be transferred, and will reduce interrupts and host processor loading. When the appropriate conditions are matched, the block will send out a DMA request.

The DMA can deal with the following conditions:

- TXFIFO empty
- TXFIFO data request
- RXFIFO data request
- RXFIFO full

The figure below shows a program sequence of SPI bursts using DMA control.



**Figure 19-13. Program Sequence of SPI Burst Using DMA**

## 19.4.8 Byte Order

The ECSPI does not support byte re-ordering in hardware.



## 19.5 Initialization

This section provides initialization information for ECSPI.

To initialize the block:

1. Clear the EN bit in ECSPI\_CONREG to reset the block.
2. Enable the clocks for ECSPI.
3. Set the EN bit in ECSPI\_CONREG to put ECSPI out of reset.
4. Configure corresponding IOMUX for ECSPI external signals.
5. Configure registers of ECSPI properly according to the specifications of the external SPI device.

## 19.6 Applications

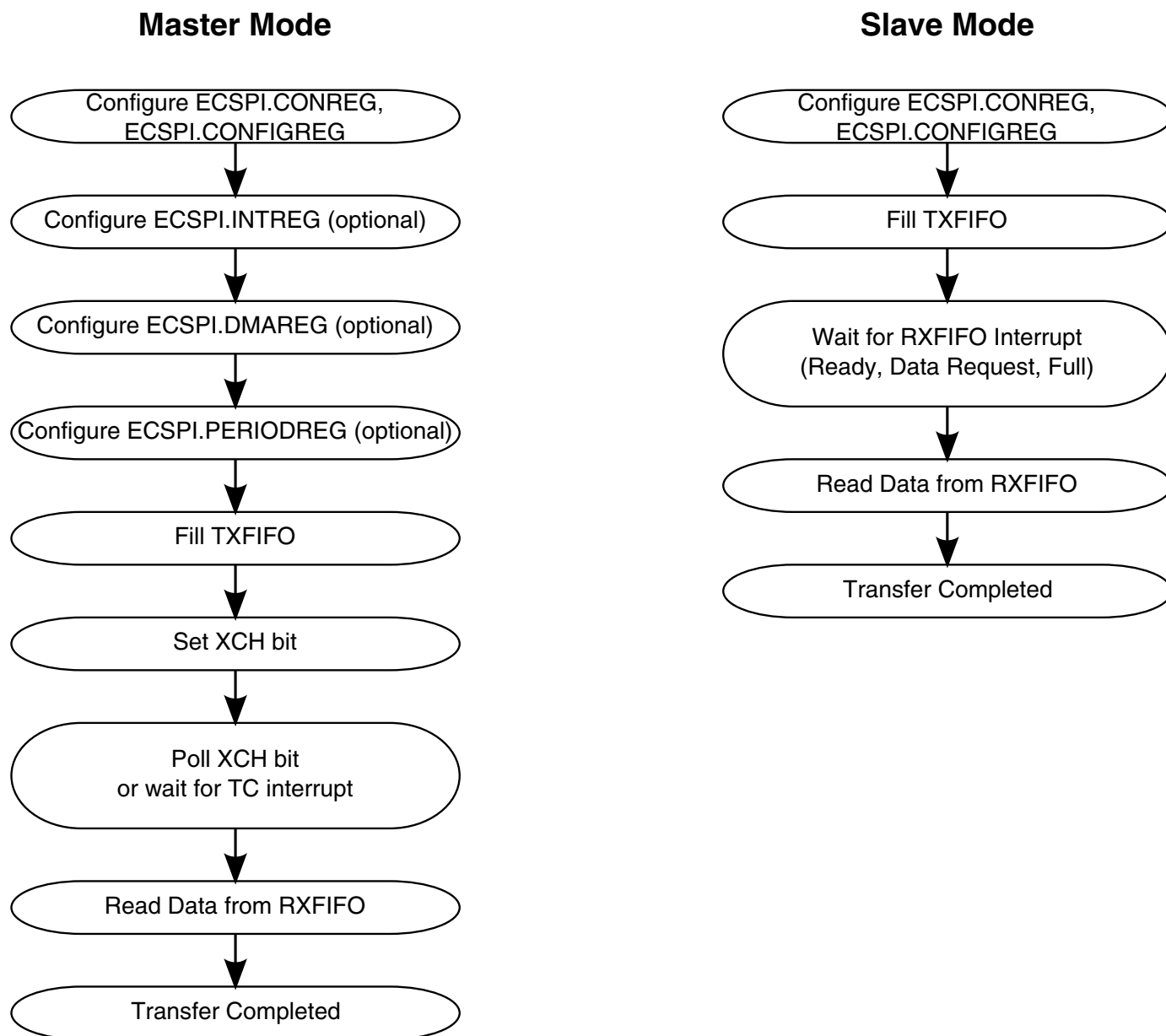


Figure 19-14. Flowchart of the ECSPI Operation

## 19.7 ECSPI Memory Map/Register Definition

This section includes the block memory map and detailed descriptions of all registers. For the base address of a particular block instantiation, see the system memory map.

## ECSPI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
200_8000	Receive Data Register (ECSPI1_RXDATA)	32	R	0000_0000h	<a href="#">19.7.1/744</a>
200_8004	Transmit Data Register (ECSPI1_TXDATA)	32	W	0000_0000h	<a href="#">19.7.2/745</a>
200_8008	Control Register (ECSPI1_CONREG)	32	R/W	0000_0000h	<a href="#">19.7.3/745</a>
200_800C	Config Register (ECSPI1_CONFIGREG)	32	R/W	0000_0000h	<a href="#">19.7.4/748</a>
200_8010	Interrupt Control Register (ECSPI1_INTREG)	32	R/W	0000_0000h	<a href="#">19.7.5/750</a>
200_8014	DMA Control Register (ECSPI1_DMAREG)	32	R/W	0000_0000h	<a href="#">19.7.6/751</a>
200_8018	Status Register (ECSPI1_STATREG)	32	R/W	0000_0003h	<a href="#">19.7.7/753</a>
200_801C	Sample Period Control Register (ECSPI1_PERIODREG)	32	R/W	0000_0000h	<a href="#">19.7.8/754</a>
200_8020	Test Control Register (ECSPI1_TESTREG)	32	R/W	0000_0000h	<a href="#">19.7.9/756</a>
200_8040	Message Data Register (ECSPI1_MSGDATA)	32	W	0000_0000h	<a href="#">19.7.10/757</a>
200_C000	Receive Data Register (ECSPI2_RXDATA)	32	R	0000_0000h	<a href="#">19.7.1/744</a>
200_C004	Transmit Data Register (ECSPI2_TXDATA)	32	W	0000_0000h	<a href="#">19.7.2/745</a>
200_C008	Control Register (ECSPI2_CONREG)	32	R/W	0000_0000h	<a href="#">19.7.3/745</a>
200_C00C	Config Register (ECSPI2_CONFIGREG)	32	R/W	0000_0000h	<a href="#">19.7.4/748</a>
200_C010	Interrupt Control Register (ECSPI2_INTREG)	32	R/W	0000_0000h	<a href="#">19.7.5/750</a>
200_C014	DMA Control Register (ECSPI2_DMAREG)	32	R/W	0000_0000h	<a href="#">19.7.6/751</a>
200_C018	Status Register (ECSPI2_STATREG)	32	R/W	0000_0003h	<a href="#">19.7.7/753</a>
200_C01C	Sample Period Control Register (ECSPI2_PERIODREG)	32	R/W	0000_0000h	<a href="#">19.7.8/754</a>
200_C020	Test Control Register (ECSPI2_TESTREG)	32	R/W	0000_0000h	<a href="#">19.7.9/756</a>
200_C040	Message Data Register (ECSPI2_MSGDATA)	32	W	0000_0000h	<a href="#">19.7.10/757</a>
201_0000	Receive Data Register (ECSPI3_RXDATA)	32	R	0000_0000h	<a href="#">19.7.1/744</a>
201_0004	Transmit Data Register (ECSPI3_TXDATA)	32	W	0000_0000h	<a href="#">19.7.2/745</a>
201_0008	Control Register (ECSPI3_CONREG)	32	R/W	0000_0000h	<a href="#">19.7.3/745</a>
201_000C	Config Register (ECSPI3_CONFIGREG)	32	R/W	0000_0000h	<a href="#">19.7.4/748</a>
201_0010	Interrupt Control Register (ECSPI3_INTREG)	32	R/W	0000_0000h	<a href="#">19.7.5/750</a>
201_0014	DMA Control Register (ECSPI3_DMAREG)	32	R/W	0000_0000h	<a href="#">19.7.6/751</a>
201_0018	Status Register (ECSPI3_STATREG)	32	R/W	0000_0003h	<a href="#">19.7.7/753</a>
201_001C	Sample Period Control Register (ECSPI3_PERIODREG)	32	R/W	0000_0000h	<a href="#">19.7.8/754</a>
201_0020	Test Control Register (ECSPI3_TESTREG)	32	R/W	0000_0000h	<a href="#">19.7.9/756</a>
201_0040	Message Data Register (ECSPI3_MSGDATA)	32	W	0000_0000h	<a href="#">19.7.10/757</a>
201_4000	Receive Data Register (ECSPI4_RXDATA)	32	R	0000_0000h	<a href="#">19.7.1/744</a>
201_4004	Transmit Data Register (ECSPI4_TXDATA)	32	W	0000_0000h	<a href="#">19.7.2/745</a>
201_4008	Control Register (ECSPI4_CONREG)	32	R/W	0000_0000h	<a href="#">19.7.3/745</a>
201_400C	Config Register (ECSPI4_CONFIGREG)	32	R/W	0000_0000h	<a href="#">19.7.4/748</a>
201_4010	Interrupt Control Register (ECSPI4_INTREG)	32	R/W	0000_0000h	<a href="#">19.7.5/750</a>
201_4014	DMA Control Register (ECSPI4_DMAREG)	32	R/W	0000_0000h	<a href="#">19.7.6/751</a>

Table continues on the next page...

## ECSPI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
201_4018	Status Register (ECSPI4_STATREG)	32	R/W	0000_0003h	<a href="#">19.7.7/753</a>
201_401C	Sample Period Control Register (ECSPI4_PERIODREG)	32	R/W	0000_0000h	<a href="#">19.7.8/754</a>
201_4020	Test Control Register (ECSPI4_TESTREG)	32	R/W	0000_0000h	<a href="#">19.7.9/756</a>
201_4040	Message Data Register (ECSPI4_MSGDATA)	32	W	0000_0000h	<a href="#">19.7.10/757</a>

## 19.7.2 Receive Data Register (ECSPiX\_RXDATA)

The Receive Data register (ECSPI\_RXDATA) is a read-only register that forms the top word of the 64 x 32 receive FIFO. This register holds the data received from an external SPI device during a data transaction. Only word-sized read operations are allowed.

Address: Base address + 0h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ECSPI_RXDATA																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## ECSPiX\_RXDATA field descriptions

Field	Description
31–0 ECSPI_RXDATA	Receive Data. This register holds the top word of the receive data FIFO. The FIFO is advanced for each read of this register. The data read is undefined when the Receive Data Ready (RR) bit in the Interrupt Control/Status register is cleared. Zeros are read when ECSPI is disabled.

### 19.7.3 Transmit Data Register (ECSPIx\_TXDATA)

The Transmit Data (ECSPI\_TXDATA) register is a write-only data register that forms the bottom word of the 64 x 32 TXFIFO. The TXFIFO can be written to as long as it is not full, even when the SPI Exchange bit (XCH) in ECSPI\_CONREG is set. This allows software to write to the TXFIFO during a SPI data exchange process. Writes to this register are ignored when the ECSPI is disabled (ECSPI\_CONREG[EN] bit is cleared).

Address: Base address + 4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W	ECSPI_TXDATA																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### ECSPIx\_TXDATA field descriptions

Field	Description
31–0 ECSPI_TXDATA	Transmit Data. This register holds the top word of data loaded into the FIFO. Data written to this register must be a word operation. The number of bits actually transmitted is determined by the BIT_COUNT field of the corresponding SPI Control register. If this field contains more bits than the number specified by BIT_COUNT, the extra bits are ignored. For example, to transfer 10 bits of data, a 32-bit word must be written to this register. Bits 9-0 are shifted out and bits 31-10 are ignored. When the ECSPI is operating in Slave mode, zeros are shifted out when the FIFO is empty. Zeros are read when ECSPI is disabled.

### 19.7.4 Control Register (ECSPIx\_CONREG)

The Control Register (ECSPI\_CONREG) allows software to enable the ECSPI , configure its operating modes, specify the divider value, and SPI\_RDY control signal, and define the transfer length.

Address: Base address + 8h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BURST_LENGTH												CHANNEL_SELECT		DRCTL	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PRE_DIVIDER				POST_DIVIDER				CHANNEL_MODE				SMC	XCH	HT	EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## ECSPiX\_CONREG field descriptions

Field	Description
31–20 BURST_LENGTH	<p>Burst Length. This field defines the length of a SPI burst to be transferred. The Chip Select (SS) will remain asserted until all bits in a SPI burst are shifted out. A maximum of <math>2^{12}</math> bits can be transferred in a single SPI burst.</p> <p>In master mode, it controls the number of bits per SPI burst. Since the shift register always loads 32-bit data from transmit FIFO, only the <math>n</math> least-significant (<math>n = \text{BURST\_LENGTH} + 1</math>) will be shifted out. The remaining bits will be ignored.</p> <p>In slave mode, only when SS_CTL is cleared, this field will take effect in the transfer.</p> <p>Number of Valid Bits in a SPI burst.</p> <p>0x000 A SPI burst contains the 1 LSB in a word.  0x001 A SPI burst contains the 2 LSB in a word.  0x002 A SPI burst contains the 3 LSB in a word.  ...  0x01F A SPI burst contains all 32 bits in a word.  0x020 A SPI burst contains the 1 LSB in first word and all 32 bits in second word.  0x021 A SPI burst contains the 2 LSB in first word and all 32 bits in second word.  ...  0xFFE A SPI burst contains the 31 LSB in first word and <math>2^7 - 1</math> words.  0xFFF A SPI burst contains <math>2^7</math> words.</p>
19–18 CHANNEL_SELECT	<p>SPI CHANNEL SELECT bits. Select one of four external SPI Master/Slave Devices. In master mode, these two bits select the external slave devices by asserting the Chip Select (SS<sub>n</sub>) outputs. Only the selected Chip Select (SS<sub>n</sub>) signal can be active at a given time; the remaining three signals will be negated.</p> <p>00 Channel 0 is selected. Chip Select 0 (SS0) will be asserted.  01 Channel 1 is selected. Chip Select 1 (SS1) will be asserted.  10 Channel 2 is selected. Chip Select 2 (SS2) will be asserted.  11 Channel 3 is selected. Chip Select 3 (SS3) will be asserted.</p>
17–16 DRCTL	<p>SPI Data Ready Control. This field selects the utilization of the <math>\overline{\text{SPI\_RDY}}</math> signal in master mode. ECSPI checks this field before it starts an SPI burst.</p> <p>00 The <math>\overline{\text{SPI\_RDY}}</math> signal is a don't care.  01 Burst will be triggered by the falling edge of the SPI_RDY signal (edge-triggered).  10 Burst will be triggered by a low level of the SPI_RDY signal (level-triggered).  11 Reserved.</p>
15–12 PRE_DIVIDER	<p>SPI Pre Divider. ECSPI uses a two-stage divider to generate the SPI clock. This field defines the pre-divider of the reference clock.</p> <p>0000 Divide by 1.  0001 Divide by 2.  0010 Divide by 3.  ...  1101 Divide by 14.  1110 Divide by 15.  1111 Divide by 16.</p>
11–8 POST_DIVIDER	<p>SPI Post Divider. ECSPI uses a two-stage divider to generate the SPI clock. This field defines the post-divider of the reference clock using the equation: <math>2^n</math>.</p> <p>0000 Divide by 1.</p>

Table continues on the next page...

**ECSPiX\_CONREG field descriptions (continued)**

Field	Description
	0001 Divide by 2. 0010 Divide by 4. ... 1110 Divide by $2^{14}$ . 1111 Divide by $2^{15}$ .
7–4 CHANNEL_ MODE	SPI CHANNEL MODE selects the mode for each SPI channel. CHANNEL MODE[3] is for SPI channel 3. CHANNEL MODE[2] is for SPI channel 2. CHANNEL MODE[1] is for SPI channel 1. CHANNEL MODE[0] is for SPI channel 0.  0 Slave mode. 1 Master mode.
3 SMC	Start Mode Control. This bit applies only to channels configured in Master mode (CHANNEL MODE = 1). It controls how the ECSPI starts a SPI burst, either through the SPI exchange bit, or immediately when the TXFIFO is written to.  0 SPI Exchange Bit (XCH) controls when a SPI burst can start. Setting the XCH bit will start a SPI burst or multiple bursts. This is controlled by the SPI SS Wave Form Select (SS_CTL). Refer to XCH and SS_CTL descriptions. 1 Immediately starts a SPI burst when data is written in TXFIFO.
2 XCH	SPI Exchange Bit. This bit applies only to channels configured in Master mode (CHANNEL MODE = 1). If the Start Mode Control (SMC) bit is cleared, writing a 1 to this bit starts one SPI burst or multiple SPI bursts according to the SPI SS Wave Form Select (SS_CTL). The XCH bit remains set while either the data exchange is in progress, or when the ECSPI is waiting for an active input if SPIRDY is enabled through DRCTL. This bit is cleared automatically when all data in the TXFIFO and the shift register has been shifted out.  0 Idle. 1 Initiates exchange (write) or busy (read).
1 HT	Hardware Trigger Enable. This bit is used in master mode only. It enables hardware trigger (HT) mode. Note, HT mode is not supported by this product.  0 Disable HT mode. 1 Enable HT mode.
0 EN	SPI Block Enable Control. This bit enables the ECSPI. This bit must be set before writing to other registers or initiating an exchange. Writing zero to this bit disables the block and resets the internal logic with the exception of the ECSPiX_CONREG. The block's internal clocks are gated off whenever the block is disabled.  0 Disable the block. 1 Enable the block.

## 19.7.5 Config Register (ECSPIx\_CONFIGREG)

The Config Register (ECSPI\_CONFIGREG) allows software to configure each SPI channel, configure its operating modes, specify the phase and polarity of the clock, configure the Chip Select (SS), and define the HT transfer length. Note, HT mode is not supported by this product.

Address: Base address + Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
R	Reserved								HT_LENGTH				SCLK_CTL				DATA_CTL				SS_POL				SS_CTL				SCLK_POL				SCLK_PHA			
W	Reserved								HT_LENGTH				SCLK_CTL				DATA_CTL				SS_POL				SS_CTL				SCLK_POL				SCLK_PHA			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

### ECSPIx\_CONFIGREG field descriptions

Field	Description
31–29 -	This field is reserved. Reserved
28–24 HT_LENGTH	HT LENGTH. This field defines the message length in HT Mode. Note, HT mode is not supported by this product. The length in bits of one message is (HT LENGTH + 1).
23–20 SCLK_CTL	SCLK CTL. This field controls the inactive state of SCLK for each SPI channel. SCLK CTL[3] is for SPI channel 3. SCLK CTL[2] is for SPI channel 2. SCLK CTL[1] is for SPI channel 1. SCLK CTL[0] is for SPI channel 0.  0 Stay low. 1 Stay high.
19–16 DATA_CTL	DATA CTL. This field controls inactive state of the data line for each SPI channel. DATA CTL[3] is for SPI channel 3. DATA CTL[2] is for SPI channel 2. DATA CTL[1] is for SPI channel 1. DATA CTL[0] is for SPI channel 0.  0 Stay high. 1 Stay low.
15–12 SS_POL	SPI SS Polarity Select. In both Master and Slave modes, this field selects the polarity of the Chip Select (SS) signal. SS POL[3] is for SPI channel 3. SS POL[2] is for SPI channel 2. SS POL[1] is for SPI channel 1. SS POL[0] is for SPI channel 0.

*Table continues on the next page...*



**ECSPiX\_CONFIGREG field descriptions (continued)**

Field	Description
	<p>0 Active low. 1 Active high.</p>
11–8 SS_CTL	<p>SPI SS Wave Form Select. In master mode, this field controls the output wave form of the Chip Select (SS) signal when the SMC (Start Mode Control) bit is cleared. The SS_CTL are ignored if the SMC bit is set.</p> <p>SS_CTL[3] is for SPI channel 3. SS_CTL[2] is for SPI channel 2. SS_CTL[1] is for SPI channel 1. SS_CTL[0] is for SPI channel 0.</p> <p>In slave mode, this bit controls when the SPI burst is completed.</p> <p>An SPI burst is completed by the Chip Select (SS) signal edges. (SSPOL = 0: rising edge; SSPOL = 1: falling edge) The RXFIFO is advanced whenever a Chip Select (SS) signal edge is detected or the shift register contains 32-bits of valid data.</p> <p>0 In master mode - only one SPI burst will be transmitted. 1 In master mode - Negate Chip Select (SS) signal between SPI bursts. Multiple SPI bursts will be transmitted. The SPI transfer will automatically stop when the TXFIFO is empty.</p> <p>0 In slave mode - an SPI burst is completed when the number of bits received in the shift register is equal to (BURST LENGTH + 1). Only the n least-significant bits (n = BURST LENGTH[4:0] + 1) of the first received word are valid. All bits subsequent to the first received word in RXFIFO are valid. 1 In slave mode - an SPI burst is completed by the Chip Select (SS) signal edges. (SSPOL = 0: rising edge; SSPOL = 1: falling edge) The RXFIFO is advanced whenever a Chip Select (SS) signal edge is detected or the shift register contains 32-bits of valid data.</p>
7–4 SCLK_POL	<p>SPI Clock Polarity Control. This field controls the polarity of the SCLK signal. See <a href="#">Figure 19-10</a> for more information.</p> <p>SCLK_POL[3] is for SPI channel 3. SCLK_POL[2] is for SPI channel 2. SCLK_POL[1] is for SPI channel 1. SCLK_POL[0] is for SPI channel 0.</p> <p>0 Active high polarity (0 = Idle). 1 Active low polarity (1 = Idle).</p>
3–0 SCLK_PHA	<p>SPI Clock/Data Phase Control. This field controls the clock/data phase relationship. See <a href="#">Figure 19-10</a> for more information.</p> <p>SCLK_PHA[3] is for SPI channel 3. SCLK_PHA[2] is for SPI channel 2. SCLK_PHA[1] is for SPI channel 1. SCLK_PHA[0] is for SPI channel 0.</p> <p>0 Phase 0 operation. 1 Phase 1 operation.</p>

## 19.7.6 Interrupt Control Register (ECSPIx\_INTREG)

The Interrupt Control Register (ECSPI\_INTREG) enables the generation of interrupts to the host processor. If the ECSPI is disabled, this register reads zero.

Address: Base address + 10h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								TCEN	ROEN	RFEN	RDREN	RREN	TFEN	TDREN	TEEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**ECSPIx\_INTREG field descriptions**

Field	Description
31–8 -	This field is reserved. Reserved
7 TCEN	Transfer Completed Interrupt enable. This bit enables the Transfer Completed Interrupt. 0 Disable 1 Enable
6 ROEN	RXFIFO Overflow Interrupt enable. This bit enables the RXFIFO Overflow Interrupt. 0 Disable 1 Enable
5 RFEN	RXFIFO Full Interrupt enable. This bit enables the RXFIFO Full Interrupt. 0 Disable 1 Enable
4 RDREN	RXFIFO Data Request Interrupt enable. This bit enables the RXFIFO Data Request Interrupt when the number of data entries in the RXFIFO is greater than RX_THRESHOLD. 0 Disable 1 Enable
3 RREN	RXFIFO Ready Interrupt enable. This bit enables the RXFIFO Ready Interrupt. 0 Disable 1 Enable

*Table continues on the next page...*

**ECSPIx\_INTREG field descriptions (continued)**

Field	Description
2 TFEN	TXFIFO Full Interrupt enable. This bit enables the TXFIFO Full Interrupt. 0 Disable 1 Enable
1 TDREN	TXFIFO Data Request Interrupt enable. This bit enables the TXFIFO Data Request Interrupt when the number of data entries in the TXFIFO is less than or equal to TX_THRESHOLD. 0 Disable 1 Enable
0 TEEN	TXFIFO Empty Interrupt enable. This bit enables the TXFIFO Empty Interrupt. 0 Disable 1 Enable

**19.7.7 DMA Control Register (ECSPIx\_DMAREG)**

The Direct Memory Access Control Register (ECSPI\_DMAREG) provides software a way to use an on-chip DMA controller for ECSPI data. Internal DMA request signals enable direct data transfers between the ECSPI FIFOs and system memory. The ECSPI sends out DMA requests when the appropriate FIFO conditions are matched.

If the ECSPI is disabled, this register is read as 0.

Address: Base address + 14h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**ECSPiX\_DMAREG field descriptions**

Field	Description
31 RXTDEN	RXFIFO TAIL DMA Request Enable. This bit enables an internal counter that is increased at each read of the RXFIFO. This counter is cleared automatically when it reaches RX DMA LENGTH. If the number of words remaining in the RXFIFO is greater than or equal to RX DMA LENGTH, a DMA request is generated even if it is less than or equal to RX_THRESHOLD.  0   Disable 1   Enable
30 -	This field is reserved. Reserved
29–24 RX_DMA_LENGTH	RX DMA LENGTH. This field defines the burst length of a DMA operation. Applies only when RXTDEN is set.
23 RXDEN	RXFIFO DMA Request Enable. This bit enables/disables the RXFIFO DMA Request.  0   Disable 1   Enable
22 -	This field is reserved. Reserved
21–16 RX_THRESHOLD	RX THRESHOLD. This field defines the FIFO threshold that triggers a RX DMA/INT request. A RX DMA/INT request is issued when the number of data entries in the RXFIFO is greater than RX_THRESHOLD.
15–8 -	This field is reserved. Reserved
7 TEDEN	TXFIFO Empty DMA Request Enable. This bit enables/disables the TXFIFO Empty DMA Request.  0   Disable 1   Enable
6 -	This field is reserved. Reserved
5–0 TX_THRESHOLD	TX THRESHOLD. This field defines the FIFO threshold that triggers a TX DMA/INT request. A TX DMA/INT request is issued when the number of data entries in the TXFIFO is greater than TX_THRESHOLD.

## 19.7.8 Status Register (ECSPIx\_STATREG)

The ECSPI Status Register (ECSPI\_STATREG) reflects the status of the ECSPI's operating condition. If the ECSPI is disabled, this register reads 0x0000\_0003.

Address: Base address + 18h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								TC	RO	RF	RDR	RR	TF	TDR	TE
W									w1c	w1c						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

**ECSPIx\_STATREG field descriptions**

Field	Description
31–8 -	This field is reserved. Reserved
7 TC	Transfer Completed Status bit. Writing 1 to this bit clears it.  0 Transfer in progress. 1 Transfer completed.
6 RO	RXFIFO Overflow. When set, this bit indicates that RXFIFO has overflowed. Writing 1 to this bit clears it.  0 RXFIFO has no overflow. 1 RXFIFO has overflowed.
5 RF	RXFIFO Full. This bit is set when the RXFIFO is full.  0 Not Full. 1 Full.
4 RDR	RXFIFO Data Request.  0 When RXTDE is set - Number of data entries in the RXFIFO is not greater than RX_THRESHOLD. 1 When RXTDE is set - Number of data entries in the RXFIFO is greater than RX_THRESHOLD or a DMA TAIL DMA condition exists. 0 When RXTDE is clear - Number of data entries in the RXFIFO is not greater than RX_THRESHOLD. 1 When RXTDE is clear - Number of data entries in the RXFIFO is greater than RX_THRESHOLD.
3 RR	RXFIFO Ready. This bit is set when one or more words are stored in the RXFIFO.  0 No valid data in RXFIFO. 1 More than 1 word in RXFIFO.
2 TF	TXFIFO Full. This bit is set when if the TXFIFO is full.

*Table continues on the next page...*

**ECSPiX\_STATREG field descriptions (continued)**

Field	Description
	0 TXFIFO is not Full. 1 TXFIFO is Full.
1 TDR	TXFIFO Data Request.  0 Number of empty slots in TXFIFO is greater than TX_THRESHOLD. 1 Number of empty slots in TXFIFO is not greater than TX_THRESHOLD.
0 TE	TXFIFO Empty. This bit is set if the TXFIFO is empty.  0 TXFIFO contains one or more words. 1 TXFIFO is empty.

**19.7.9 Sample Period Control Register (ECSPiX\_PERIODREG)**

The Sample Period Control Register (ECSPiX\_PERIODREG) provides software a way to insert delays (wait states) between consecutive SPI transfers. Control bits in this register select the clock source for the sample period counter and the delay count indicating the number of wait states to be inserted between data transfers.

The delay counts apply only when the current channel is operating in Master mode (ECSPiX\_CONREG[CHANNEL MODE] = 1). ECSPiX\_PERIODREG also contains the CSD CTRL field used to insert a delay between the Chip Select's active edge and the first SPI Clock edge.

Address: Base address + 1Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved										CSD_CTL					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CSDC	SAMPLE_PERIOD														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**ECSPiX\_PERIODREG field descriptions**

Field	Description
31-22 -	This field is reserved. Reserved

Table continues on the next page...

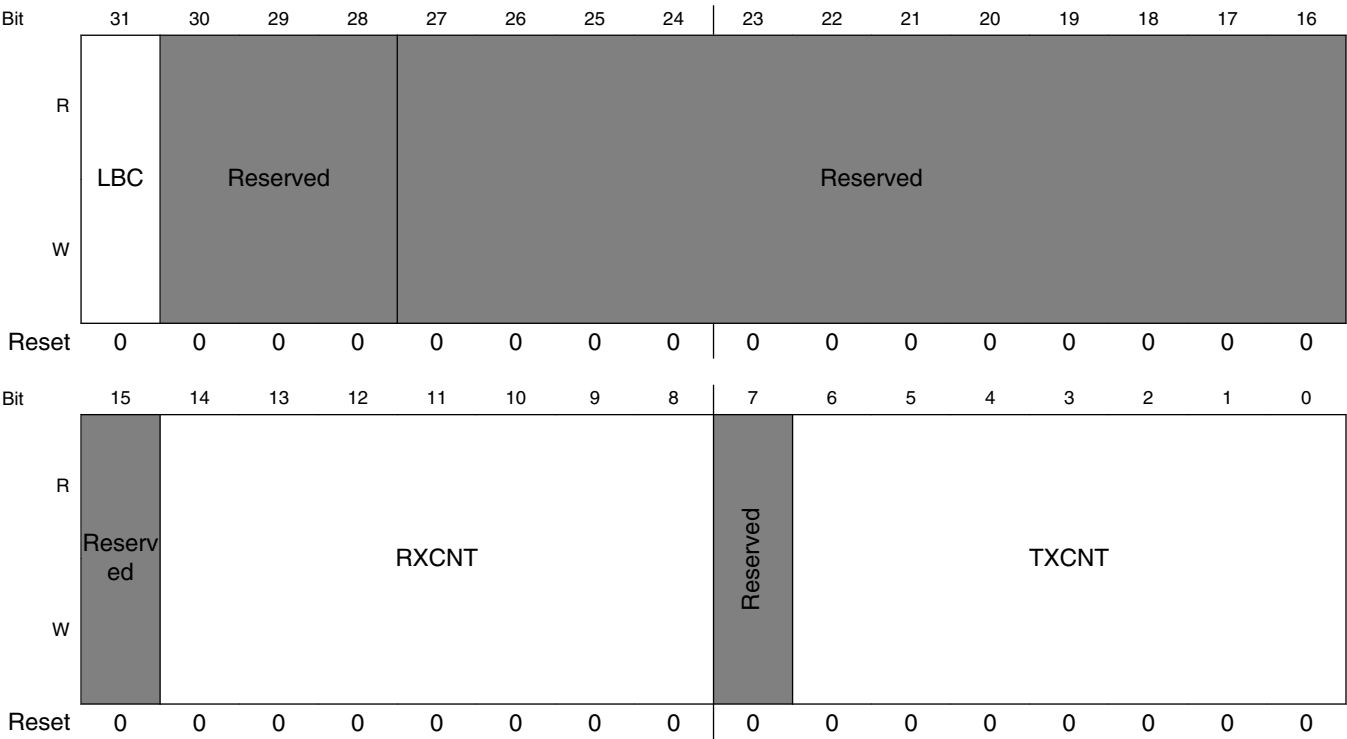
**ECSPiX\_PERIODREG field descriptions (continued)**

Field	Description
21–16 CSD_CTL	Chip Select Delay Control bits. This field defines how many SPI clocks will be inserted between the chip select's active edge and the first SPI clock edge. The range is from 0 to 63.
15 CSRC	Clock Source Control. This bit selects the clock source for the sample period counter.  0    SPI Clock (SCLK) 1    Low-Frequency Reference Clock (32.768 KHz)
14–0 SAMPLE_ PERIOD	Sample Period Control. These bits control the number of wait states to be inserted in data transfers. During the idle clocks, the state of the SS output will operate according to the SS_CTL control field in the ECSPiX_CONREG register.  0x0000    0 wait states inserted 0x0001    1 wait state inserted ...        ... 0x7FFE    32766 wait states inserted 0x7FFF    32767 wait states inserted

19.7.10 Test Control Register (ECSPIdx\_TESTREG)

The Test Control Register (ECSPI\_TESTREG) provides software a mechanism to internally connect the receive and transmit devices of the ECSPI, and monitor the contents of the receive and transmit FIFOs.

Address: Base address + 20h offset



ECSPIdx\_TESTREG field descriptions

Field	Description
31 LBC	Loop Back Control. This bit is used in Master mode only. When this bit is set, the ECSPI connects the transmitter and receiver sections internally, and the data shifted out from the most-significant bit of the shift register is looped back into the least-significant bit of the Shift register. In this way, a self-test of the complete transmit/receive path can be made. The output pins are not affected, and the input pins are ignored.  0 Not connected. 1 Transmitter and receiver sections internally connected for Loopback.
30–28 -	This field is reserved. Reserved, all bits should be ignored.
27–15 -	This field is reserved. Reserved
14–8 RXCNT	RXFIFO Counter. This field indicates the number of words in the RXFIFO.

Table continues on the next page...



**ECSPiX\_TESTREG field descriptions (continued)**

Field	Description
7 -	This field is reserved. Reserved
6–0 TXCNT	TXFIFO Counter. This field indicates the number of words in the TXFIFO.

**19.7.11 Message Data Register (ECSPiX\_MSGDATA)**

The Message Data Register (ECSPiX\_MSGDATA) forms the top word of the 16 x 32 MSG Data FIFO. Only word-size accesses are allowed for this register. Reads to this register return zero, and writes to this register store data in the MSG Data FIFO.

Address: Base address + 40h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W	ECSPiX_MSGDATA																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**ECSPiX\_MSGDATA field descriptions**

Field	Description
31–0 ECSPiX_MSGDATA	ECSPiX_MSGDATA holds the top word of MSG Data FIFO. The MSG Data FIFO is advanced for each write of this register. The data read is zero. The data written to this register is stored in the MSG Data FIFO.





## **Chapter 20**

# **External Interface Module (EIM)**

### **20.1 Overview**

The EIM handles the interface to devices external to the chip, including generation of chip selects, clock and control for external peripherals and memory. It provides asynchronous access to devices with SRAM-like interface and synchronous access to devices with NOR-Flash-like or PSRAM-like interface.

## Overview

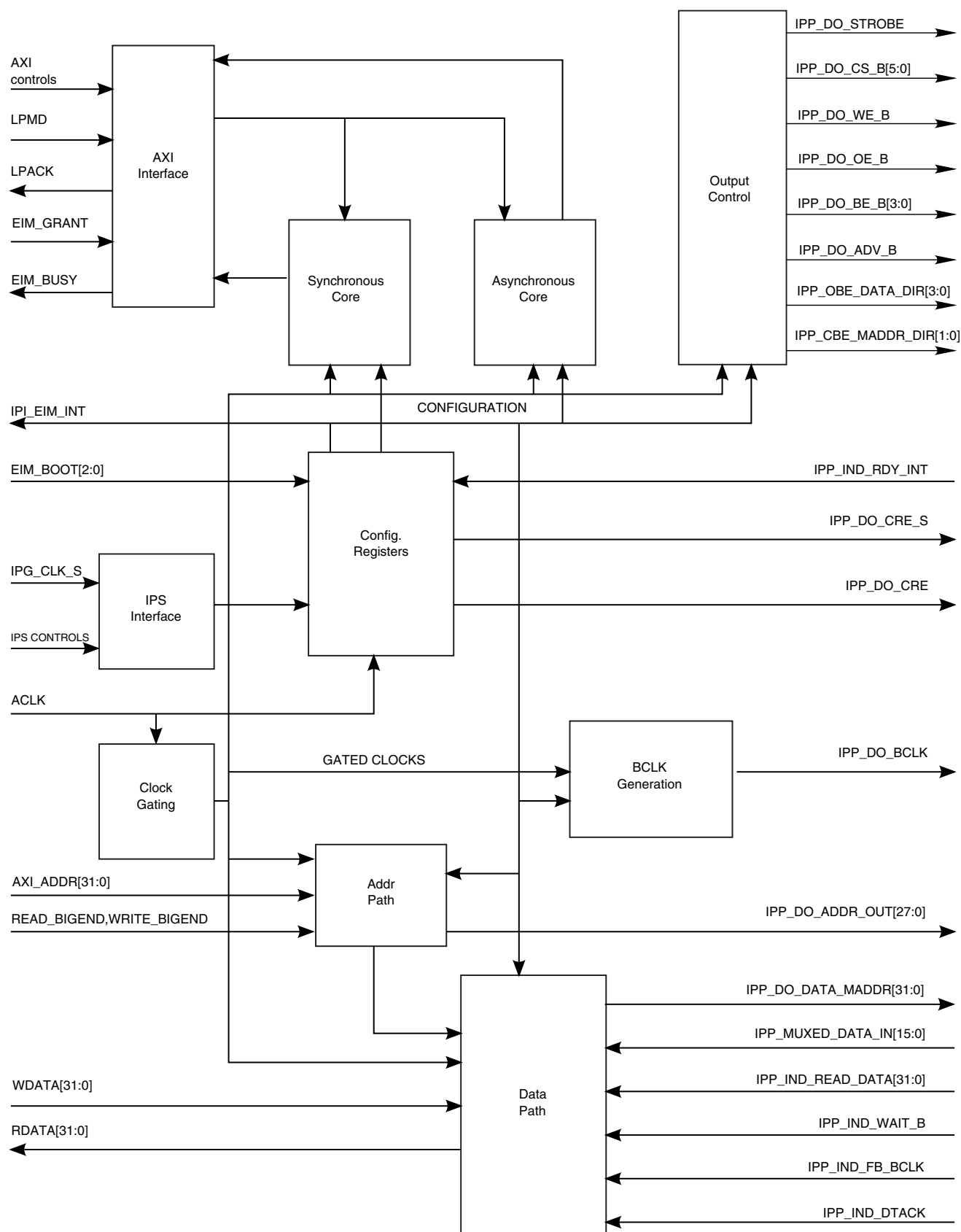


Figure 20-1. EIM Diagram

## 20.1.1 Features

- Up to six chip selects for external devices
  - Flexible address decoding. Each chip select memory space determined separately, according to VIA port configuration (see [Chip Select Memory Map](#)). Configurable Chip Select 0 base address (by VIA)
  - Individual select signal for each one of the memory space defined. Up to 6 memory spaces may be defined and programmed individually.
  - 26-bit external address bus, max memory size can be 128 MByte (1 Gigabit).
- Selectable Write Protection for each Chip Select
- Support for multiplexed address / data bus operation x16 and x32 port size
- Programmable Data Port Size for each Chip Select (x8, x16 and x32)
- Programmable Wait-State generator for each Chip Select, for write and read accesses separately
- Asynchronous accesses with programmable setup and hold times for control signals
- Support for Asynchronous page mode accesses (x16 and x32 port size)
- Support continuous Burst Clock which can be used as reference clock for FPGA
- Independent synchronous Memory Burst Read Mode support for NOR-Flash and PSRAM memories (x16 and x32 port size)
- Independent synchronous Memory Burst Write Mode support for PSRAM and NOR-Flash like memories (CellularRAM™ from Micron, Infineon, and Cypress, OneNAND™ and utRAM™ from Samsung, and COSMORAM™ from Toshiba)
- Support of NAND-Flash devices with NOR-Flash like interface - MDOC™ (M-Systems), OneNAND™ (Samsung)
- Independent programmable variable/fix Latency support for read and write synchronous (burst) mode
- Support for Big Endian and Little Endian operation modes per access
- ARM AXI slave interface. One ID at a time support.
- External Interrupt support, RDY\_INT signal function as external interrupt
- Boot from external device support according to boot signals, using RDY\_INT signal
  - RDY signal support assertion after reset for MDOC™ (M-Systems) device
  - INT signal support assertion after reset for OneNAND™ (Samsung) device

## 20.1.2 Modes of Operation

The EIM has the following modes of operation:

- Asynchronous Mode
- Asynchronous Page Mode
- Multiplexed Address/Data mode

- Burst Clock Mode
- Low Power Modes
- Boot Mode

See details in the [EIM Operational Modes](#).

### 20.1.2.1 Asynchronous Mode

This is a non-burst mode that is used for SRAM access. In this mode, a single data is read/written with each access (asserted address).

All controls' timings are controlled by preset values in Chip Select Configuration Registers.

### 20.1.2.2 Asynchronous Page Read Mode

Setting the APR bit causes the EIM to perform memory burst accesses by emulating page mode operation.

The external address asserts for each piece of data. The initial access timing is according to RWSC field, and the next address assertions timing is according to PAT field. When APR bit is set, RCSN OEN, RADVN and RBEN fields are ignored for burst access to the external device.

The page size can be set via the BL field to 2, 4, 8, 16, or 32 words (the word size is determined by the DSZ field).

### 20.1.2.3 Multiplexed Address/Data Mode

In this mode, multiplexing addresses and data bits on the same pins is supported for synchronous/asynchronous accesses to x8/x16/ x32 data width memory devices.

For more information about the pins that drive data/address in 8/16/32 non-muxed mode and 16/32 muxed mode, refer to the EIM Internal Module Multiplexing table in the EIM Internal Pads Allocation chapter of the datasheet.

**Table 20-1. EIM multiplexing**

Setup	Non Multiplexed Address/Data Mode							Multiplexed Address/ Data mode	
	8 Bit				16 Bit		32 Bit	16 Bit	32 Bit
	MUM = 0, DSZ = 100	MUM = 0, DSZ = 101	MUM = 0, DSZ = 110	MUM = 0, DSZ = 111	MUM = 0, DSZ = 001	MUM = 0, DSZ = 010	MUM = 0, DSZ = 011	MUM = 1, DSZ = 001	MUM = 1, DSZ = 011
A[15:0]	EIM_DA[15:0]	EIM_DA[15:0]	EIM_DA[15:0]	EIM_DA[15:0]	EIM_DA[15:0]	EIM_DA[15:0]	EIM_DA[15:0]	EIM_DA[15:0]	EIM_DA[15:0]
A[25:16]	EIM_A[25:16]	EIM_A[25:16]	EIM_A[25:16]	EIM_A[25:16]	EIM_A[25:16]	EIM_A[25:16]	EIM_A[25:16]	EIM_A[25:16]	EIM_D[9:0]
D[7:0], EIM_EB0	EIM_D[7:0]	-	-	-	EIM_D[7:0]	-	EIM_D[7:0]	EIM_DA[7:0]	EIM_DA[7:0]
D[15:8], EIM_EB1	-	EIM_D[15:8]	-	-	EIM_D[15:8]	-	EIM_D[15:8]	EIM_DA[15:8]	EIM_DA[15:8]
D[23:16], EIM_EB2	-	-	EIM_D[23:16]	-	-	EIM_D[23:16]	EIM_D[23:16]	-	EIM_D[7:0]
D[31:24], EIM_EB3	-	-	-	EIM_D[31:24]	-	EIM_D[31:24]	EIM_D[31:24]	-	EIM_D[15:8]

#### 20.1.2.4 Burst Clock Mode

The controller has the ability to support burst synchronous operations in various frequencies, depending on the frequency of the input clock supplied by the system (EIM clock).

The EIM clock can be divided by one, two, three or four, and its frequency can be changed according to the requirements. Variable and fix latency are supported for this mode, according to the external device requirements.

- Synchronous read mode. This is a burst mode, which is used for reading from Flash/PSRAM memory devices. In this mode, after address assertion a burst of sequential data can be read. Data exchange is carried out according to BCLK being generated by EIM. An access is delayed according to external WAIT\_B signal assertion (signal from the memory device).
- Synchronous write mode. A burst mode used for accessing external devices, which support synchronous write type of access (PSRAM protocol). In this mode, after address assertion a burst of sequential data can be written to the external device. Access may be delayed according to WAIT\_B signal assertion (signal from the memory device) before first piece of data arrived to the external device.

#### NOTE

Maximum frequency of the EIM main clock is 133Mhz. It may be reduced by the system for special cases of external devices,

which demand a different frequency than integer division of the 133MHz clock.

### 20.1.2.5 Low Power Modes

The input clock is gated by ACT\_CS bits. When all the ACT\_CS are negated (all CS disable) the internal clock is turned off; awready/wready & arready signal are de-asserted and the master can't access the EIM.

### 20.1.2.6 Boot Mode

It is possible to perform a boot operation from external device located on CS0. The configuration of the relevant bits are done with boot mode signals according to the external device parameters (for example, port size and protocol assertion).

See for more details.

## 20.2 External Signals

The following table describes the external signals of EIM:

**Table 20-2. EIM External Signals**

Signal	Description	Pad	Mode	Direction
EIM_ACLK_FREERUN	AXI clock signal	EPDC_PWRINT	ALT3	I
EIM_AD00	LSB multiplexed Address/Data Bus signal	KEY_COL0	ALT3	IO
EIM_AD01	LSB multiplexed Address/Data Bus signal	KEY_ROW0	ALT3	IO
EIM_AD02	LSB multiplexed Address/Data Bus signal	KEY_COL1	ALT3	IO
EIM_AD03	LSB multiplexed Address/Data Bus signal	KEY_ROW1	ALT3	IO
EIM_AD04	LSB multiplexed Address/Data Bus signal	KEY_COL2	ALT3	IO
EIM_AD05	LSB multiplexed Address/Data Bus signal	KEY_ROW2	ALT3	IO
EIM_AD06	LSB multiplexed Address/Data Bus signal	KEY_COL3	ALT3	IO
EIM_AD07	LSB multiplexed Address/Data Bus signal	KEY_ROW3	ALT3	IO

*Table continues on the next page...*



**Table 20-2. EIM External Signals (continued)**

Signal	Description	Pad	Mode	Direction
EIM_AD08	LSB multiplexed Address/Data Bus signal	KEY_COL4	ALT3	IO
EIM_AD09	LSB multiplexed Address/Data Bus signal	KEY_ROW4	ALT3	IO
EIM_AD10	LSB multiplexed Address/Data Bus signal	KEY_COL5	ALT3	IO
EIM_AD11	LSB multiplexed Address/Data Bus signal	KEY_ROW5	ALT3	IO
EIM_AD12	LSB multiplexed Address/Data Bus signal	KEY_COL6	ALT3	IO
EIM_AD13	LSB multiplexed Address/Data Bus signal	KEY_ROW6	ALT3	IO
EIM_AD14	LSB multiplexed Address/Data Bus signal	KEY_COL7	ALT3	IO
EIM_AD15	LSB multiplexed Address/Data Bus signal	KEY_ROW7	ALT3	IO
EIM_ADDR16	MSB Address Bus signal	EPDC_D8	ALT3	O
EIM_ADDR17	MSB Address Bus signal	EPDC_D9	ALT3	O
EIM_ADDR18	MSB Address Bus signal	EPDC_D10	ALT3	O
EIM_ADDR19	MSB Address Bus signal	EPDC_D11	ALT3	O
EIM_ADDR20	MSB Address Bus signal	EPDC_D12	ALT3	O
EIM_ADDR21	MSB Address Bus signal	EPDC_D13	ALT3	O
EIM_ADDR22	MSB Address Bus signal	EPDC_D14	ALT3	O
EIM_ADDR23	MSB Address Bus signal	EPDC_D15	ALT3	O
EIM_ADDR24	MSB Address Bus signal	EPDC_VCOM0	ALT3	O
EIM_ADDR25	MSB Address Bus signal	EPDC_VCOM1	ALT3	O
EIM_ADDR26	MSB Address Bus signal	EPDC_BDR0	ALT3	O
EIM_BCLK	Burst Clock (BCLK). This active-high output signal is used to clock external burstcapable devices to synchronize the loading and incrementing of addresses and delivery of burst read and write data to/from the EIM. Its behavior is affected by the BCM field in the EIM_WCR and the SWR, SRD, BCD, and BCS fields of the EIM_CSxGCR1.	EPDC_PWRCOM	ALT3	O
EIM_CRE	Used as CRE/PS for CellularRam memory. It is used for the Mode Register Set command. This signal can be configured as active low or active high. See CRE and CREP field descriptions of the EIM_CSxGCR1 registers.	EPDC_BDR1	ALT3	O

*Table continues on the next page...*

**Table 20-2. EIM External Signals (continued)**

Signal	Description	Pad	Mode	Direction
EIM_CS0	Chip Selects. These signals are active-low. Behavior is affected by the RCSA and RCSN fields of the EIM_CSxRCR1 registers and the WCSA and WCSN fields of the EIM_CSxWCR1 registers.	EPDC_PWRCTRL2	ALT3	O
		LCD_HSYNC	ALT3	
EIM_CS1	Chip Selects. These signals are active-low. Behavior is affected by the RCSA and RCSN fields of the EIM_CSxRCR1 registers and the WCSA and WCSN fields of the EIM_CSxWCR1 registers.	EPDC_PWRCTRL3	ALT3	O
		LCD_VSYNC	ALT3	
EIM_CS2	Chip Selects. These signals are active-low. Behavior is affected by the RCSA and RCSN fields of the EIM_CSxRCR1 registers and the WCSA and WCSN fields of the EIM_CSxWCR1 registers.	EPDC_SDCE0	ALT3	O
EIM_CS3	Chip Selects. These signals are active-low. Behavior is affected by the RCSA and RCSN fields of the EIM_CSxRCR1 registers and the WCSA and WCSN fields of the EIM_CSxWCR1 registers.	LCD_DAT5	ALT3	O
EIM_DATA00	MSB Data Bus signal	LCD_DAT6	ALT3	IO
EIM_DATA01	MSB Data Bus signal	LCD_DAT7	ALT3	IO
EIM_DATA02	MSB Data Bus signal	LCD_DAT8	ALT3	IO
EIM_DATA03	MSB Data Bus signal	LCD_DAT9	ALT3	IO
EIM_DATA04	MSB Data Bus signal	LCD_DAT10	ALT3	IO
EIM_DATA05	MSB Data Bus signal	LCD_DAT11	ALT3	IO
EIM_DATA06	MSB Data Bus signal	LCD_DAT12	ALT3	IO
EIM_DATA07	MSB Data Bus signal	LCD_DAT13	ALT3	IO
EIM_DATA08	MSB Data Bus signal	LCD_DAT14	ALT3	IO
EIM_DATA09	MSB Data Bus signal	LCD_DAT15	ALT3	IO
EIM_DATA10	MSB Data Bus signal	LCD_DAT16	ALT3	IO
EIM_DATA11	MSB Data Bus signal	LCD_DAT17	ALT3	IO
EIM_DATA12	MSB Data Bus signal	LCD_DAT18	ALT3	IO
EIM_DATA13	MSB Data Bus signal	LCD_DAT19	ALT3	IO
EIM_DATA14	MSB Data Bus signal	LCD_DAT20	ALT3	IO
EIM_DATA15	MSB Data Bus signal	LCD_DAT21	ALT3	IO
EIM_DTACK_B	Data Acknowledge, asynchronous access. This input is used as a data acknowledge signal for single asynchronous accesses.	EPDC_PWRWAKEUP	ALT3	I
		LCD_RESET	ALT1	

*Table continues on the next page...*

**Table 20-2. EIM External Signals (continued)**

Signal	Description	Pad	Mode	Direction
EIM_EB0	<p>Byte Enable. These active-low output signals indicate valid data bytes for the current access. They may be configured to assert for write cycles only.</p> <p>EIM_EB[0] corresponds to DATA_OUT[7:0]</p> <p>For asynchronous write accesses, behavior is affected by the WBEA and WBEN fields of the EIM_CS1WCR1-EIM_CS5WCR1 Registers. On synchronous or asynchronous read accesses, these signals are always asserted at the start of the access and negated at end of the access.</p>	EPDC_SDCE2	ALT3	O
EIM_EB1	<p>Byte Enable. These active-low output signals indicate valid data bytes for the current access. They may be configured to assert for write cycles only.</p> <p>EIM_EB[1] corresponds to DATA_OUT[15:8]</p> <p>For asynchronous write accesses, behavior is affected by the WBEA and WBEN fields of the EIM_CS1WCR1-EIM_CS5WCR1 Registers. On synchronous or asynchronous read accesses, these signals are always asserted at the start of the access and negated at end of the access.</p>	EPDC_SDCE3	ALT3	O
EIM_EB2	<p>Byte Enable. These active-low output signals indicate valid data bytes for the current access. They may be configured to assert for write cycles only.</p> <p>EIM_EB[2] corresponds to DATA_OUT[23:16]</p> <p>For asynchronous write accesses, behavior is affected by the WBEA and WBEN fields of the EIM_CS1WCR1-EIM_CS5WCR1 Registers. On synchronous or asynchronous read accesses, these signals are always asserted at the start of the access and negated at end of the access.</p>	LCD_DAT23	ALT3	O

*Table continues on the next page...*

**Table 20-2. EIM External Signals (continued)**

Signal	Description	Pad	Mode	Direction
EIM_EB3	Byte Enable. These active-low output signals indicate valid data bytes for the current access. They may be configured to assert for write cycles only.  EIM_EB[3] corresponds to DATA_OUT[31:24].  For asynchronous write accesses, behavior is affected by the WBEA and WBEN fields of the EIM_CS1WCR1-EIM_CS5WCR1 Registers. On synchronous or asynchronous read accesses, these signals are always asserted at the start of the access and negated at end of the access.	LCD_DAT22	ALT3	O
EIM_LBA	Address Valid. This active-low output signal is asserted during burst mode accesses to cause the external burst capable device to load a new starting burst address. Assertion of LBA indicates that a valid address is present on the address bus. Its behavior is affected by the SWR, SRD, BCD, and BCS fields of the EIM_CSxGCR1 registers, the RADVA and RADVN fields of the EIM_CSxRCR1 registers, and the WADVA and WADVN fields of the EIM_CSxWCR1 registers. In asynchronous mode, LBA length is affected by the RADVA, WADVA, RADVN, and WADVN fields. Minimum length of LBA signal in all modes is one EIM clock cycle.	EPDC_SDCE1	ALT3	O
EIM_OE	Output Enable. This active-low output signal indicates the bus access is a read and enables external devices to drive the data bus with read data. Its behavior is affected by the OEA and OEN bit fields in the Chip Select Configuration Registers.	EPDC_PWRCTRL1	ALT3	O
		LCD_ENABLE	ALT3	
EIM_RW	Memory Write Enable. This active-low output signal indicates the bus access is a write and enables external devices to sample the data bus. Its behavior is affected by the WEA and WEN bit fields in the Chip Select Configuration Registers.	EPDC_PWRCTRL0	ALT3	O
		LCD_CLK	ALT3	

*Table continues on the next page...*

**Table 20-2. EIM External Signals (continued)**

Signal	Description	Pad	Mode	Direction
EIM_WAIT	Ready/Busy/Wait signal. This active-low input signal is asserted by external burst capable devices which support fixed or variable latency of data. It is serviced in synchronous mode only (EIM_CSxGCR1[SWR, SRD] =1). WAIT will have a pull up resistor in I/O. The signal indicates whether the External device is ready for data transaction or not. Busy cycles (or wait cycles) of the external device can occur at the start of a Burst access or at page boundary crossover.  <b>NOTE:</b> For burst devices, WAIT output should be configured to change one cycle before data is ready (before delay).  <b>NOTE:</b> Some External devices may not use this input signal for ready state indication (fix latency without WAIT signal monitoring). For these devices EIM should be configured accordingly (see RFL, WFL, and PSZ field descriptions).  <b>NOTE:</b> This is same as what is shown in IP_IND_WAIT_B	EPDC_PWRSTAT	ALT3	I
		LCD_RESET	ALT3	

### 20.2.1 Other Important Block I/O Signals Internal to the SoC

The following table provides a description of other signals which are internal to the that are important to understand the function of EIM.

Name	I/O	Description
EIM_FB_BCLK	Input	Burst Clock Feedback. This block input is used to sample read data during high transfer speeds. The signal provides feedback from the I/O pad of the BCLK output pin and tends to align more closely with data from the external memory device.
EIM_BOOT	Input	EIM Boot Configuration. These block inputs determine the reset state of DSZ[1:0] and MUM. See Table 5-4 for detailed description.
ACLK	Input	AXI clock, maximum frequency 133 Mhz
IPG_CLK_S	Input	EIM module IPG clock
RST_B	Input	Active low HW reset
EIM_WARM_RESET	Input	Warm Reset. If this signal is asserted the rst_b will reset only the internal FF and state machine while S/W registers will keep their current state. This signal is active high signal.

## 20.3 Clocks

The following table describes the clock sources for EIM. Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 20-4. EIM Clocks**

Clock name	Clock Root	Description
aclk	aclk_eim_slow_clk_root	EIM clock (main)
aclk_slow	aclk_eim_slow_clk_root	EIM clock (slow)
ipg_clk_s	ipg_clk_root	Peripheral access clock
aclk_exsc	aclk_eim_slow_clk_root	EIM clock (external device)

- **ACLK:** EIM clock (main clock, AXI clock) with a Max frequency of 133Mhz. Can be gated externally when there is no active AXI access.
- **ACLK\_SLOW:** EIM all time running ACLK. Used for flip-flops that must be active even when EIM is in low power down mode to provide clock for lpack/lpmd registers, IP registers and IP to AXI sync registers.
- **IPG\_CLK\_S:** IPG clock for IP accesses. IP registers are activated by ACLK\_SLOW clock.
- **ACLK\_EXSC:** Clock created from EIM clock for External device usage. Integer division by 1, 2, 3 and 4 of the clock can be use with BCD bit field configuration, according to external devices demands. EIM clock frequency may be reduced for lower frequency support which cannot be achieved via BCD bit field.

## 20.4 Chip Select Memory Map

**Table 20-5. EIM Chip Select Memory Map**

Address	Space Size	Use	Access
EIM_NFC_BASE/ACT_CS/ADDRS0 inputs	128MB	CS0 memory region	R/W
ACT_CS/ADDRS1 inputs	128MB	CS1 memory region	R/W
ACT_CS/ADDRS2 inputs	128MB	CS2 memory region	R/W
ACT_CS/ADDRS3 inputs	128MB	CS3 memory region	R/W

## 20.5 Functional Description

This section provides the functional description for the EIM.

### 20.5.1 Continuous BCLK

EIM module can be programmed to provide a continuous BCLK in synchronous mode as the reference clock of PPGA. In this case, an internal DLL(Delay-locked Loop) is used to synchronize the internal clock which is used to sample read data from the I/O pad, with the feedback BCLK from PAD. The DLL compares the phase difference of internal clock to the feedback BCLK and will lock up when the delay of Reference Delay Line equals to the phase difference. After the lock up of Reference Delay Chain , DLL will copy the Master Delay Chain's value to Slave Delay Chain automatically. The Slave Delay Chain is used to delay the internal clock. If continuous BCLK is not selected, DLL should not be enabled.

When using an continuous BCLK in synchronous mode, BCD MUST be set to 0, otherwise will cause malfunction of EIM.

To let EIM work properly under continuous BCLK MODE, the initialization must follow below procedure. The initialize procedure should be done before any access to EIM. If it is not possible to do so, e.g. device booted up from EIM, the initialize flow should be done after performing a reset of EIM.

The recommended initialize flow is as follow

1. Disable EIM clock by clearing bit 4 of EIM\_WIAR Register.
2. Select Continuous BCLK by setting bit 3 of EIM\_WCR Register.
3. Enable DLL by setting bit 0 of EIM\_DCR Register.
4. Enable EIM clock by setting bit 4 of EIM\_WIAR Register.
5. Reset DLL by toggling bit 1 of EIM\_DCR Register(1->0->1).
6. Wait for DLL lock (Both bit 0 and bit 1 of EIM\_DSR Register are asserted).

After initialization, EIM can be programed to various access timing, but BCD MUST be kept to 0.

DLL Upadte Interval and inital value can be set by write EIM\_DCR Register in corresponding bit field.

Also DLL provides manual adjustment to delay. An extra offset can be added to internal delay line, which can be used when DLL have some kind of predicable error. OVERRIDE bit field of EIM\_DCR register will override reference DLL locked value and use a direct value for delay.

Note: During initialization, BCLK frequency and duty cycle are not guaranteed.

## 20.5.2 Bus Sizing Configuration

The EIM supports byte, half word and word operands allowing access to x8, x16, x32 ports. It can be address/data multiplexed in x16, x32 ports. The port size is programmable via the DSZ bit field in the corresponding Chip Select Configuration Register. An 8-bit port can reside in each one of the bytes of the data bus. A 16-bit port can reside on the lower 16 bits of the data bus, DATA\_IN/OUT[15:0] or on the higher 16 bits of the data bus, DATA\_IN/OUT[31:16].

In the case of a multi-cycle transfer, the lower two address bits (ADDR[1:0]) are incremented appropriately. The EIM address bus is configured according to DSZ bit field and AUS bits. There is either one bit (for x16 port size) or two bits (for x32 port size) right shift of the address bits (only when AUS=0) and no bit shift when AUS = 1 or DSZ[2] = 1.

The EIM has a data multiplexer which takes the four bytes of the AXI data bus and routes them to their required positions to properly interface to memory.

### NOTE

A word access to or from a x16 port requires two external bus cycles to complete the transfer.

A word access to or from a x8 port requires four external bus cycles to complete the transfer.

### 20.5.2.1 8 BIT PORT SUPPORT

EIM has limited support for mot68000 & intel 386 protocols.

#### 20.5.2.1.1 MOTOROLA 68000

EIM has limited support for mot68000 protocol. Only basic read or write asynchronous operations are supported.

The following operations are not supported:

- Read modify write
- Sync access
- All special accesses (ARM platform space, bus arbitration, bus control, bus error & reset operations)
- FC outputs



### 20.5.2.1.2 INTEL 386

EIM has limited support for intel 386 protocol. Only basic read or write async non-pipelined operations are supported.

The following operations are not supported:

- Other bus cycles (interrupt, halt & refresh)
- Bus lock
- M/IO, DC, LBA, NA, REFRESH & BS8 signals

## 20.5.3 EIM Operational Modes

Listed here are the main operational modes for EIM selected by control bit fields settings.

For details, see the bit field descriptions of SWR / SRD / MUM. All modes are supported in with 8-, 16- or 32-bit port configuration, according to DSZ bit field.

**Table 20-6. EIM Operation Modes Field Settings**

Control bit fields			Brief mode description
MUM	SRD	SWR	
0	0	0	Asynchronous write / Asynchronous read for APR=0 / Asynchronous page read for APR=1, none multiplexed
		1	Synchronous write/ Asynchronous read or APR=0 / Asynchronous page read for APR=1,none multiplexed
	1	0	Asynchronous write/Synchronous read none multiplexed
		1	Synchronous write/read none multiplexed
	1	0	0
1			Synchronous write/ Asynchronous read multiplexed
1		0	Asynchronous write/Synchronous read multiplexed
		1	Synchronous write/read multiplexed

## 20.5.4 Burst Mode (Synchronous) Memory Operation

This mode is enabled for read or write access. Bit SWR sets the burst mode for write operations at the corresponding chip select and bit SRD sets it for read operation.

When this mode is set, the controller attempts to translate the Master burst accesses to memory burst accesses, being limited by the memory burst length, predefined by BL value, or memory and Master WRAP/INCR boundary crossing non-matching. Only the first address accessed is put by the controller on the external address bus in a memory burst sequence.

EIM may translate from some Master sequential accesses to one or several memory bursts, but not from two Master individual accesses to one memory burst.

For the first access in a memory burst sequence, the EIM asserts  $\overline{ADV}$ , causing the external burst device to latch the starting burst address; then toggle the burst clock (BCLK) for a predefined number of cycles in order to latch the first unit of data. Subsequent accessed data units can then be burst in fewer clock cycles, realizing an overall increase in bus bandwidth.

### **NOTE**

The BCLK signal toggles only when burst access is executed toward the external device (BCM=1'b0 for normal mode use). It runs with a 50% duty cycle until the end of access is reached. When access is terminated, BCLK stops toggling.

Memory burst accesses are terminated by the EIM whenever it detects the following:

- The specific burst length has executed completely (end of access)
- Write access - missing data in write buffer (Master is delaying the data transfer toward the EIM)
- Next sequential access crosses boundary with unequal condition (wrap/increment, burst length) on the Master and memory
- Current memory burst length reached

## **20.5.5 Burst Clock Divisor (BCD)**

In some cases, it may be necessary to slow the external bus in relation to the internal bus to allow accesses to burst devices that have a maximum operating frequency less than the operating frequency of the internal bus.

The internal bus frequency can be divided by one, two, three or four for presentation on the external bus in burst mode operation.

BCLK can only be set to integer divisions of the incoming clock frequency. To get a specific frequency on BCLK, configure the divider to change the incoming EIM clock accordingly.

By programming the BCD bit field to various values, two signals on the external bus are affected;  $\overline{ADV}$  and BCLK. The  $\overline{ADV}$  signal is asserted according to RADVA or WADVA bit fields programming, and is negated according to the formula mentioned in RADVN and WADVN bit fields description. The BCLK signal runs with a 50% duty cycle until the end of access is reached.

If BCM = 1, the BCLK runs at frequency according to GBCD bit field settings on every async memory access, regardless of the SWR and SRD bits configuration. Caution should be exercised when using BCM bit; GBCD bit field should be updated once and should not change when BCLK is toggling. The BCM bit is used mainly for system debug mode. It has no functional use of the EIM in normal mode.

### 20.5.6 Burst Clock Start (BCS)

In an effort to allow greater flexibility in achieving the minimum number of wait states on burst accesses, you can determine when you want the BCLK to start toggling after the start of access. This allows the BCLK to be skewed from point of data capture on the EIM clock by any number of EIM clock cycles.

Care must be exercised when setting BCS bit field in conjunction with the BCD and RWSC/WWSC bit fields. See the external timing diagrams in [Burst \(Synchronous Mode\) Read Memory Accesses Timing Diagram - BCD=1](#) and [Burst \(Synchronous Mode\) Read Memory Accesses Timing Diagram - BCD=0](#) for examples of how to use the BCS, BCD and RWSC/WWSC bit fields together.

### 20.5.7 Multiplexed Address/Data Mode Support

The control bit MUM allows support memory with multiplexed address/data bus both in asynchronous and in synchronous modes.

Caution should be exercised for using OEA/WEA & ADH bit fields. They should be configured according to the external device requirements, as it determines the time point of end of address phase and start of data phase.

### 20.5.8 Mixed Master/Memory Burst Modes Support

To provide mixed sequential/wrap accesses with different length, EIM interprets burst signal and generate additional  $\overline{ADV}$  signals whenever there appear unequal address or burst boundary crossing condition.

BL bit field is used to notify EIM about current memory burst and wrap condition for properly external address generation. In case of non-matching boundaries in both the memory and Master access, EIM starts a new memory burst access by updating address from Master on address bus and generating  $\overline{ADV}$  signal.

## 20.5.9 AXI (Master) Bus Cycles Support

The EIM uses an ARM AXI slave interface. It has a 32-bit bus and supports one access (one ID) at a time. No out of order or parallel accesses are supported.

The following AXI protocol signals are not supported:

- AWLOCK
- AWCACHE
- ARLOCK
- ARCACHE

ARID bus is sampled when:

- new read access is valid on the read address channel and is reflected on the RID bus output toward the master.

AWID bus is sampled when:

- new write access is valid on the write address channel and is reflected on the WID/BID bus output toward the master.

ARPROT and AWPROT signal are partially used. ARPROT[0] and AWPROT[0] bits are used for normal/privileged access detection. ARPROT[2:1] and AWPROT[2:1] are not used.

When sampling a valid access on both of the address channels, the read access will be performed first while write access is pending. After last data transfer completed, the pending write will be executed.

A new access may be executed one cycle after sampling a valid access on the read or write address channels, assuming there is no current access (back to back) which can cause a recovery or end of access penalty cycles, for write access, also assuming data is in write buffer for fast execution.

### NOTE

- Only 32-bit word size accesses are supported for burst mode accesses.
- Only 8-bit (1 byte), 16-bit (2 byte) and 32-bit (4 byte) word size supported for single access.

- Maximum number of burst length is 16.
- According to AXI protocol, burst access should not cross 4 KB blocks. In case EIM gets an access that crosses the 4 KB, memory address calculation is invalid.

AXI transfers shown in the table below are also supported. These AXI cycles will be translated into the necessary cycles on the memory side. For example, for optimal operation in case ARM cache is configured to 8 beat burst with wrap, a synchronous flash and cellular RAM memory should be configured in 16 word wrap burst mode when using a 16-bit data port, and in 8 word wrap burst mode when using a 32-bit data port. EIM uses BL bit field to support different memory configurations. The controller splits the transaction when needed in some cases. See [Table 20-8](#).

**Table 20-7. AXI Burst Cycles Supported**

Burst Length - Number of data transfers	Burst size - Bytes in transfer	Burst type	Description
1	1	INCR	Single transfer
1	2	INCR	Single transfer
1	4	INCR	Single transfer
2	4	WRAP	2-beat wrapping burst
4	4	WRAP	4-beat wrapping burst
8	4	WRAP	8-beat wrapping burst
16	4	WRAP	16-beat wrapping burst
2	4	INCR	2-beat incrementing burst
3	4	INCR	3-beat incrementing burst
4	4	INCR	4-beat incrementing burst
5	4	INCR	5-beat incrementing burst
6	4	INCR	6-beat incrementing burst
7	4	INCR	7-beat incrementing burst
8	4	INCR	8-beat incrementing burst
9	4	INCR	9-beat incrementing burst
10	4	INCR	10-beat incrementing burst
11	4	INCR	11-beat incrementing burst
12	4	INCR	12-beat incrementing burst
13	4	INCR	13-beat incrementing burst
14	4	INCR	14-beat incrementing burst
15	4	INCR	15-beat incrementing burst
16	4	INCR	16-beat incrementing burst

**Table 20-8. AXI to Memory Burst Splits Number**

AXI Burst Type	Memory Burst Type Config.	# of accesses to X8 Memory Port size	# of accesses to X16 Memory Port size	# of accesses to X32 Memory Port size
INC16 Aligned Addr.	WRAP4	16	8	4
	Cont.	1	1	1
INC16 Unaligned Addr.	WRAP4	17	9	5
	Cont.	1	1	1
WRAP16 Aligned Addr.	WRAP16	4	2	1
	Cont.	1	1	1
WRAP16 Unaligned Addr.	WRAP16	5	3	1
	Cont.	2	2	2
INC8 Aligned Addr.	WRAP8	4	2	1
	WRAP16	2	1	1
INC8 Unaligned Addr.	WRAP8	4 or 5	2 or 3	2
	WRAP16	2 or 3	2	1 or 2
WRAP8 Aligned Addr.	WRAP16	2	1	1
	Cont.	1	1	1
WRAP8 Unaligned Addr.	WRAP16	2 or 3	1	2
	Cont.	2	2	2

### 20.5.10 WAIT\_B Signal, RWSC and WWSC bit fields Usage

Most of the external devices supporting burst mode for write or read accesses provide a signal which indicates data is valid on the memory bus (a.k.a. handshake mode). For this mode, RFL and WFL bits should be cleared and RWSC/ WWSC bit fields indicate when the controller should start sampling this signal from the external device or, in other words, how many BCLK cycles should be masked.

For devices which do not use this signal or have a fixed latency ability, the RFL and WFL bits may be set for internal calculation regarding BCLK cycles penalty until data is valid (memory initial access time). For this mode, RWSC/ WWSC indicates when the data is ready for sampling by the controller (read access) or the external device (write access). There is separation between read and write accesses wait-state control. For read access, RWSC bit field is valid and WWSC bit field is ignored; for write access, WWSC is valid and RWSC is ignored.

### 20.5.11 IPS Register Interface

Access to the registers of the EIM, read or write, is made with IPS protocol signals. The system should avoid changing the registers while master/memory transaction is valid, as this can cause an unknown behavior of the controller.

Register access size is 32-bit as the register size definition, other size of access (byte or half word) is not supported.

### 20.5.12 MRS Set for PSRAM

Memory registers of PSRAM devices can be configured according to external signal, which indicates whether the access is to a memory array or memory register domain.

When the CRE bit is set, the following transactions to the external device will assert the CRE signal. The polarity of this signal is determined by the CREP bit for active low or active high assertion of the signal.

### 20.5.13 EIM Access Termination

EIM is monitoring the corresponding CSx control signal every time variable latency access or dtack access is performed toward the external device.

In variable latency accesses, the Watchdog Timer (WDOG-1) counts BCLK cycles. If it reaches the wdog\_limit (according to the WDOG\_LIMIT bit field in the WCR) before the device signals can drive/sample new data, the controller will terminate the access and generate an error response transfer toward the Master.

In dtack access, WDOG-1 counts ACLK cycles instead of BCLK and it reaches the wdog\_limit before the device asserts the dtack signal, the controller will terminate the access and generate an error response transfer toward the Master.

WDOG-1 can be disabled by WDOG\_EN bit in the WCR.

### 20.5.14 Error Conditions

The following conditions cause an error (AXI error or IPS error) response signal:

- AXI errors
  - Access to a disabled chip select - access to a mapped chip select address space where the CSEN bit in the corresponding chip select Configuration Register is clear

- Access to a non mapped address - access to an address that is not mapped to any CS.
- User access to a supervisor-protected chip select address space (the SP bit in the corresponding chip select Configuration Register is set)
- User access in fixed mode access
- User performs write access to write protected chip select
- First write data ID and write address ID do not match. (No data is written to the memory.)
- First Write Data ID and write address ID match but one or more of the other Write data IDs does not match the First Write data ID (data is written to memory according)
- Access duration to external device from CSx signal assertion is 128/256/512/1024 cycles (access is terminated by the controller) - This error can be disabled by software.
- IPS errors
  - User read or write access to a reserved/non-valid address in the EIM Configuration Register

## 20.5.15 DTACK Mode

In DTACK mode, the EIM uses DTACK signal as an indication of when to end the access.

DTACK is an asynchronous edge/level sensitive signal. DTACK polarity is configurable by the DAP bit in CsxGCR2 (default value is 0).

In this case, EIM begins the access and after a few cycles (according DAPS field) and waits until DTACK (after synchronization) becomes asserted, then samples the data in read access and completes the current data access (see [Figure 20-15](#), [Figure 20-16](#) & [Figure 20-17](#)).

If more than one data is needed, CS will be negated between access (CSREC field is not zero) and the AXI burst access will be split into single accesses (see [Figure 20-19](#)).

## 20.5.16 RDY\_INT Signal as Interrupt

The EIM has an external interrupt support. When INTEN bit in the WCR is set, signal RDY\_INT is used as interrupt; its status is being reflected by INT bit and output signal.



It is cleared by writing one to the INT bit. When INTEN is cleared, the interrupt is disabled. This interrupt is a level interrupt and its polarity can be configured by the INTPOL bit in the WCR.

### 20.5.17 RDY\_INT Signal as Ready After Reset Indication

This feature is used for boot propose from external devices based on NANDFlash array memory with NORFlash interface.

When ERRST bit is set, RDY\_INT signal is monitored to determine ready after reset of the external device located on CS0.

The monitoring is taking place when CS0 is accessed for the first time. The access will be pending until assertion of the signal is detected. When detection occurs, ERRST bit is self-cleared and pending access is executed to the external device on CS0.

### 20.5.18 EIM\_GRANT / EIM\_BUSY Handshake Description

Prior to executing command to one of the external device (chip select), EIM assert EIM\_BUSY signal (1'b1) and checks the EIM\_GRANT signal status.

If EIM\_GRANT signal is high, it indicates external data bus is not used by other slaves (NAND Flash Controller) and EIM may start to execute the access. If EIM\_GRANT is low, EIM waits until it is set (1'b1) before executing the access.

EIM keeps EIM\_BUSY signal set until it completes the access toward the external device.

Once EIM\_GRANT signal is set, it can not be reset until EIM\_BUSY signal is cleared by EIM.

#### NOTE

In 16-bit Muxed EIM doesn't use the data bus, therefore there is no sharing of the data bus with NFC. EIM doesn't wait for EIM\_GRANT signal from NFC and doesn't assert the EIM\_BUSY signal.

## 20.5.19 LPMD / LPACK Handshake Description

These signals are used for frequency and/or voltage change, and for entering low power mode during normal operation of the EIM. Before any change can take place, the controller and all the relevant external devices should be in idle state, which means no access or data transfer is in process.

LPMD input signal is asserted once EIM detects the assertion of LPMD, all ready signals of the AXI channels are negated, and EIM is not sampling new accesses. It finishes all the ongoing accesses and already pending ones. When EIM is in idle state, the LPACK output signal is asserted. EIM will stay in idle state and the LPACK signal will stay asserted until the LPMD signal is negated.

## 20.5.20 Endianness

Big and Little endianness are supported by the controller according to the following table.

**Table 20-9. EIM Out/in Data in Case AXI Out/in Data is 0xB3B2B1B0**

Endian mode	AXI access	AXI address [1:0]	Port size and used bits								
			Word port				Half word port			Byte port	
			[31:24]	[23:16]	[15:8]	[7:0]	External address [0]	[31:24] ([15:8])	[23:16] ([7:0])	External address [1:0]	[31:24] ([23:16]) ([15:8]) ([7:0])
Big	Word	0	0xB3	0xB2	0xB1	0xB0	0	0xB3	0xB2	0	0xB3
										1	0xB2
							1	0xB1	0xB0	2	0xB1
										3	0xB0
	Half Word	0			0xB1	0xB0	0	0xB3	0xB2	0	0xB3
										1	0xB2
		2	0xB3	0xB2			1	0xB1	0xB0	2	0xB1
										3	0xB0
	Byte	0				0xB0	0		0xB3	0	0xB3
								1			0xB1
		2		0xB2			1		0xB1	2	0xB1
								3	0xB3		

Table continues on the next page...

**Table 20-9. EIM Out/in Data in Case AXI Out/in Data is 0xB3B2B1B0 (continued)**

Endian mode	AXI access	AXI address [1:0]	Port size and used bits								
			Word port				Half word port			Byte port	
			[31:24]	[23:16]	[15:8]	[7:0]	External address [0]	[31:24] ([15:8])	[23:16] ([7:0])	External address [1:0]	[31:24] ([23:16]) ([15:8]) ([7:0])
Little	Word	0	0xB3	0xB2	0xB1	0xB0	0	0xB1	0xB0	0	0xB0
							1			0xB1	
							1	0xB3	0xB2	2	0xB2
							3			0xB3	
	Half Word	0			0xB1	0xB0	0	0xB1	0xB0	0	0xB0
							1			0xB1	
		2	0xB3	0xB2			1	0xB3	0xB2	2	0xB2
							3			0xB3	
	Byte	0				0xB0	0		0xB0	0	0xB0
		1			0xB1			0xB1		1	0xB1
		2		0xB2			1		0xB2	2	0xB2
		3	0xB3					0xB3		3	0xB3

## 20.5.21 Strobe Signal Use

The strobe signal is toggling according to address/data valid condition on the external bus for read and write accesses, and for both synchronous and asynchronous modes.

At any time point when address/data is valid on the external bus, the strobe signal will generate a positive edge, which can be used to sample the external data and control signal.

### NOTE

Strobe signal for read data is active (RL + 1) cycles after data on external bus is valid.

## 20.6 Initialization Information

## 20.6.1 Booting from EIM

EIM is ready to work with CS0 after the hardware reset, but it has been configured for very slow access (for boot purposes), with additional setup and hold time.

Other CSs are disabled by hardware reset. Therefore, all CSs must be properly initialized before use in writing values to the corresponding chip select configuration registers.

DSZ[1:0] and MUM fields are set according to EIM\_BOOT [2:0] block inputs.

## 20.7 Typical Application

Application note uses following functions to illustrate EIM and memory accesses:

- WR16(address, data) is a 16 bit write access
- WR32(address, data) is a 32 bit write access
- RD16(address, data) is a 16 bit read access
- RD32(address, data) is a 32 bit read access
- WR\_I(address, data, delta, counter) is a write data sequence, there  $\text{data}(i+1) = \text{data}(i) + \text{delta}$
- COMMAND\_SEQUENCE
- CHECK\_STATUS

### NOTE

COMMAND\_SEQUENCE and CHECK\_STATUS are described in [AMD Flash Utility](#), [Intel Sibley Flash Utility](#), [MDOC Device Utility](#), [Samsung OneNAND Utility](#), and [Spansion Flash Utility](#).

All addresses are byte addresses. "CS0" is a Chip Select 0 base address. "EIM\_" is a prefix of EIM's registers. 'h' is a prefix of hexadecimal constant. "///" is a comment beginning. csba[cs] is a dimension of CS base addresses. "addr" means an address offset in current CS address space. Examples use CS0 address space, but it may apply to any CS except for boot mode functionality.

Configuration examples were verified with the memory models listed below and may require some adjustments for other family members.

### 20.7.1 Access to AMD Flash

The following mentioned configurations are intended for AMD Simultaneous Flash Memory w/ VersatileIO™ Control.

### 20.7.1.1 AMD Flash Asynchronous Mode Configuration

EIM register configuration used for AMD flash asynchronous access:

- WR32('EIM\_CS0RCR1,'h0a018000);
- WR32('EIM\_CS0WCR1,'h0704a240);
- WR32('EIM\_CS0GCR1,'h00430081); // for 32 bit memory
- WR32('EIM\_CS0GCR1,'h00410081); // for 16 bit memory

### 20.7.1.2 AMD Flash Utility

```
// Single data word programming to addr, 32-bit memory
port_size = 32;
WR32('CS0+('h0555<<2), 'h00aa); // command sequence
WR32('CS0+('h02aa<<2), 'h0055);
WR32('CS0+('h0555<<2), 'ha0); // command code
WR32('CS0+addr, data); // addr & data
// Polling end of programming
edata = data; // expected data
data = ~edata;
errst = 0;
while(!(data == edata) &&!errst)
begin: BR_EN
RD16('CS0+addr, data); // Read status
if(data[7] != edata[7])
begin
if(data[5] == 1)
begin
RD16('CS0+addr, data3);
RD16('CS0+addr, data);
if(data[6] != data3[6])
begin
$display("CHECK_STATUS: Error timeout on single data program");
errst = 1;
disable BR_EN;
end
end
end
else
begin
RD16('CS0+addr, data3);
if(port_size == 32)
RD32('CS0+addr, data);
else
begin
RD16('CS0+addr, data);
edata[31:16] = 16'h0;
end
end
if(data != edata)
begin
$display("CHECK_STATUS: Error in data write on single data program");
errst = 2;
disable BR_EN;
end
end
end
// Single data word programming to addr, 16-bit memory
port_size = 16;
WR32('CS0+('h0555<<1), 'h00aa); // command sequence
```

```
WR32('CS0+('h02aa<<1), 'h0055);
WR32('CS0+('h0555<<1), 'ha0); // command code
WR32('CS0+addr, data); // addr & data
// Polling end of programming - the same as for 32 bit
```

## 20.7.2 Access to Intel Sibley Flash

The following configurations are intended to Sibley family muxed and non-muxed devices.

### 20.7.2.1 Intel Sibley Flash Asynchronous Mode Configuration

- WR32('EIM\_CS0GCR1, 'h00210081);
- WR32('EIM\_CS0RCR1, 'h0e020000);
- WR32('EIM\_CS0RCR2, 'h00000000);
- WR32('EIM\_CS0WCR1, 'h0704a040);

### 20.7.2.2 Intel Sibley Flash Synchronous Mode Configuration

Configuration used for 133 MHz synchronous access to flash:

```
// Set memory to synchronous read mode
WR16('CS0+('h5903<<1), 'h0060);
WR16('CS0+('h5903<<1), 'h0003);
WR16('CS0+('h0000<<1), 'h00ff);
// Set EIM configuration to synchronous timing
WR32('EIM_CS0GCR1, 'h50214225); // 133 MHz
WR32('EIM_CS0RCR1, 'h0c000000); // 12 cycles on memory
```

Configuration used for 66 MHz synchronous access to muxed flash:

```
// Set memory to synchronous read mode
WR16('CS0+('h3103<<1), 'h0060);
WR16('CS0+('h3103<<1), 'h0003);
WR16('CS0+('h0000<<1), 'h00ff);
//-----
// Set EIM configuration to synchronous timing
WR32('EIM_CS0GCR1, 'h5021122d); // 66 MHz
WR32('EIM_CS0RCR1, 'h07000000); // 7cycles on memory
```

### 20.7.2.3 Intel Sibley Flash Utility

```
// Single data word programming to addr
WR16('CS0+addr, 'h0060); // Unlock
WR16('CS0+addr, 'h00d0);
WR16('CS0+addr, 'h0041);
WR16('CS0+addr, data);
WR16('CS0+caddr, 'h0070); // Read Status command
while('CS0+data[7] == 0) // Wait / Polling
    RD16('CS0+addr, data); // Read status
RD16('CS0+addr, data); // Read status
```

```

    WR16('CS0+'h0000,'h00ff);
// Write buffer programming
    WR16('CS0+addr,'h0060);           // Unlock
    WR16('CS0+addr,'h00d0);
    data = 0;
    WR16('CS0+addr, 'h0070);           // Read Status command
    while(data[7] == 0)                 // Wait
        RD16('CS0+addr, data);         // Read status
    WR16('CS0+'h0000,'h00ff);
    WR16('CS0+addr,'h00e9);           // Write Buffer command
    WR16('CS0+addr,255);              // Word counter (<256)
    for(i=0; i<'h200; i = i + 'h40)
        WR_I('CS0+addr+i, data+((i>2)*'h0010_0001), 'h0010_0001, 16); // Data
    WR16('CS0+addr,'h00d0);           // Write Confirm command
    data = 0;
    while(data[7] == 0)                 // Wait
        RD16('CS0+addr, data);         // Read status
    RD16('CS0+addr, data);             // Read status
    WR16('CS0+'h0000,'h00ff);

```

### 20.7.3 Access to MDOC Device

The following configurations are intended to MDOC H3 device.

#### 20.7.3.1 MDOC Device Boot

To boot from the MDOC device the ERRST bit should be configured to 1, so that EIM will hold the first read access to CS0 until the MDOC asserts the RDY signal.

#### 20.7.3.2 MDOC Device Asynchronous Mode Configuration

```

// Non-muxed mode
WR32('EIM_CS0GCR1,'h00410081);
WR32('EIM_CS0RCR1,'h0e121010);
WR32('EIM_CS0RCR2,'h00000000);
WR32('EIM_CS0WCR1,'h12092492);
// Muxed mode
WR32('EIM_CS0GCR1,'h00410081);
WR32('EIM_CS0RCR1,'h0e121010);
WR32('EIM_CS0RCR2,'h00000000);
WR32('EIM_CS0WCR1,'h12092492);

```

#### 20.7.3.3 MDOC Device Utility

```

// Read Manufacturer ID and Device ID
    RE16('CS0+'h9400,'h4833);
    RE16('CS0+'h9422,'hb7cc);

```

## 20.7.4 Access to Micron PSRAM

The following configurations are intended to mt45w4mw16bfb\_706.

### 20.7.4.1 Micron PSRAM Asynchronous Mode Configuration

```
// 16 bit memory
WR32('EIM_CS0GCR1','h403104b1');
WR32('EIM_CS0RCR1','h0b010000');
WR32('EIM_CS0RCR2','h00000008');
WR32('EIM_CS0WCR1','h0b040040');
// 32 bit memory
WR32('EIM_CS0GCR1','h403304b1');
WR32('EIM_CS0RCR1','h0f010000');
WR32('EIM_CS0RCR2','h00000008');
WR32('EIM_CS0WCR1','h0f040040');
```

### 20.7.4.2 Micron PSRAM Synchronous Mode Configuration

```
// 16 bit memory
WR32('EIM_CS0GCR1','h403104b1');
WR32('EIM_CS0WCR1','h0b040000');
WR16('CS0+('h85947<<1),'h0040'); // memory configuration
WR32('EIM_CS0GCR1','h4021_5487'); // fixed latency memory wrap 4
WR32('EIM_CS0RCR1','h04000000');
WR32('EIM_CS0RCR2','h00000008');
WR32('EIM_CS0WCR1','h04000000');
// 32 bit memory
WR32('EIM_CS0GCR1','h6003_04f1');
WR32('EIM_CS0WCR1','h0b04_0000');
WR32('CS0+('h85947<<2),'h0040'); // memory configuration
WR32('EIM_CS0GCR1','h4003_1487'); // var latency memory inc. page size 128
WR32('EIM_CS0RCR1','h04000000');
WR32('EIM_CS0RCR2','h00000008');
WR32('EIM_CS0WCR1','h04000000');
```

## 20.7.5 Access to Samsung OneNAND

Mentioned below are the configurations intended for Samsung OneNAND muxed and non-muxed devices.

### 20.7.5.1 Samsung OneNAND Boot

There are two ways to boot from Samsung OneNAND. In the first way, the ERRST bit is set to 0 and the user has to poll the interrupt status in the OneNAND interrupt register (or set interrupt handler there). In the second way, the ERRST bit is set to 1 and the user should enable the device interrupt output before the first read from CS0 access is issued.

Load sectors 2,3 to DataRAM, page 0 done in the next example:



- WR16('CS0+('hF241<<1),'h0); // Clear interrupt status
- WR16('CS0+('hF100<<1),'h0); // block[8:0] address
- WR16('CS0+('hF107<<1),'h2); // sector[1:0] and page[7:2] addresses
- WR16('CS0+('hF200<<1),'h802); // buffer[11:8] address and counter[1:0]
- WR16('CS0+('hF101<<1),'h0); // DDP choose
- WR16('CS0+('hF220<<1),'h0); // Set command

### 20.7.5.2 Samsung OneNAND Asynchronous Mode Configuration

```
// Non-muxed memory
WR32('EIM_CS0GCR1,'h00410081);
WR32('EIM_CS0RCR1,'h0b010000);
WR32('EIM_CS0RCR2,'h00000000);
WR32('EIM_CS0WCR1,'h0c092480);
// Muxed memory
WR32('EIM_CS0GCR1,'h00410089);
WR32('EIM_CS0RCR1,'h0b010000);
WR32('EIM_CS0RCR2,'h00000000);
WR32('EIM_CS0WCR1,'h0c092480);
```

### 20.7.5.3 Samsung OneNAND Synchronous Mode Configuration

Set memory and EIM to synchronous read mode is shown in the next example:

```
WR16('CS0+('hF221<<1),'hc0e0); // Synchronous read, 4 clk latency
WR32('EIM_CS0GCR1,'h50412405); // 44 MHz (non-muxed)
WR32('EIM_CS0RCR1,'h05010000);
```

The muxed Samsung OneNAND supports synchronous write, too:

```
// Set memory & EIM to synchronous read and write mode
WR16('CS0+('hF221<<1),'hc0f2); // Sync. read and write, 4 clk latency
WR32('EIM_CS0GCR1,'h5041240f); // 44 MHz
WR32('EIM_CS0RCR1,'h05010000);
WR32('EIM_CS0WCR1,'h05040000);
```

### 20.7.5.4 Samsung OneNAND Utility

The following utility algorithms are used on the Samsung OneNAND:

```
// Unlock Block command
WR16('CS0+('hF100<<1),'h0); // DFS
WR16('CS0+('hF100<<1),'h0); // DBS
WR16('CS0+('hF24c<<1),'h2); // SBA - block number (2)
WR16('CS0+('hF241<<1),'h0); // Clear interrupt status
WR16('CS0+('hF220<<1),'h23); // Unlock command
data = 'h0;
while(!(data & 'h0004)) // Polling
    RD32('WIAR, data); // Read status
// Erase block command
WR16('CS0+('hF100<<1),'h2); // DFS and block ([8:0]) address
WR16('CS0+('hF101<<1),'h0); // DBS
WR16('CS0+('hF241<<1),'h0); // Clear interrupt status
```

## Typical Application

```
WR16('CS0+('hF220<<1), 'h94); // Erase command
data = 'h0;
while(!(data &'h0004)) // Wait
    RD32('WIAR, data); // Read status
// Program page command
WR16('CS0+('hF100<<1), 'h2); // DFS and block[8:0] address
WR16('CS0+('hF107<<1), 'h0); // sector[1:0] and page[7:2] addresses
WR16('CS0+('hF200<<1), 'h800); // buffer[11:8] address and counter[1:0]
WR16('CS0+('hF241<<1), 'h0); // Clear interrupt status
WR16('CS0+('hF220<<1), 'h80); // Program command
data = 'h0;
while(!(data &'h0004)) // Wait
    RD32('WIAR, data); // Read status
```

## 20.7.6 Access to Samsung UtRAM

Below mentioned configurations are intended for Samsung UtRAM.

### 20.7.6.1 Samsung UtRAM Asynchronous Mode Configuration

```
WR32('EIM_CS0GCR1, 'h400104b1);
WR32('EIM_CS0RCR1, 'h0a010000);
WR32('EIM_CS0RCR2, 'h00000008);
WR32('EIM_CS0WCR1, 'h0b040040);
```

### 20.7.6.2 Samsung UtRAM Synchronous Mode Configuration

```
RD16('CS0+('hff_ffff<<1), data); // command sequence
RD16('CS0+('hff_ffff<<1), data);
RD16('CS0+('hff_ffff<<1), data);
RD16('CS0+('hff_feff<<1), data);
RD16('CS0+('h00_82a0<<1), data); // memory sync. configuration
WR32('EIM_CS0GCR1, 'h4021_53b7); // fixed latency memory wrap 32
WR32('EIM_CS0RCR1, 'h0500_0000);
WR32('EIM_CS0WCR1, 'h0300_0000);
```

## 20.7.7 Access to Spansion Flash

Below mentioned configurations are intended for Spansion Flash.

### 20.7.7.1 Spansion Flash Asynchronous Mode Configuration

```
WR32('EIM_CS0GCR1, 'h00410081);
WR32('EIM_CS0RCR1, 'h0a018000);
WR32('EIM_CS0RCR2, 'h00000000);
WR32('EIM_CS0WCR1, 'h0704a240);
WR16('CS0+('hF220<<1), 'h94); // Erase command
data = 'h0;
while(!(data &'h0004)) // Wait
    RD32('WIAR, data); // Read status
// Program page command
```

```

WR16('CS0+('hF100<<1),'h2);           // DFS and block[8:0] address
WR16('CS0+('hF107<<1),'h0);           // sector[1:0] and page[7:2] addresses
WR16('CS0+('hF200<<1),'h800);         // buffer[11:8] address and counter[1:0]
WR16('CS0+('hF241<<1),'h0);           // Clear interrupt status
WR16('CS0+('hF220<<1),'h80);           // Program command
data = 'h0;
while(!(data &'h0004))                 // Wait
    RD32('WIAR, data); // Read status

```

### 20.7.7.2 Spansion Flash Synchronous Mode Configuration

```

WR16('CS0+('h0555<<1),'h00aa); // command sequence
WR16('CS0+('h02aa<<1),'h0055);
WR16('CS0+('h0555<<1),'hd0);
WR16('CS0+('h0000<<1),'h1ec4); // memory sync. configuration
WR32('EIM_CS0GCR1,'h50411325); // 66 MHz
WR32('EIM_CS0RCR1,'h05000000); // 5 cycles on memory

```

### 20.7.7.3 Spansion Flash Utility

```

// Single word programming
COMMAND_SEQUENCE(cs,16,'ha0);           // single word programming
WR16('CS0+addr, data);
CHECK_STATUS('CS0+addr,data,16,1,errst);
// Write buffer programming
COMMAND_SEQUENCE(0,16,'h25);           // write buffer programming
WR16('CS0+addr,'h001f);                 // counter-1
WR_I('CS0+addr, data, 'h0010_0001, 16); // data
WR16('CS0+addr,'h0029);                 // write buffer to flash
CHECK_STATUS('CS0+addr+'h3e,data[31:16]+'h00f0,16,1,errst);

```

There `COMMAND_SEQUENCE` and `CHECK_STATUS` are next functions:

```

task COMMAND_SEQUENCE;
    input[2:0]    cs;
    input[7:0]    port_size;
    input[31:0]   code;
begin
    if(port_size == 16)
        begin
            WR16(csba[cs]+('h0555<<1),'h00aa);
            WR16(csba[cs]+('h02aa<<1),'h0055);
            WR16(csba[cs]+('h0555<<1),code);
        end
    else
        begin
            WR32(csba[cs]+('h0555<<2),'h00aa);
            WR32(csba[cs]+('h02aa<<2),'h0055);
            WR32(csba[cs]+('h0555<<2),code);
        end
end
endtask
task CHECK_STATUS;
    input[31:0] addr;
    input[31:0] edata;
    input[7:0]   port_size;
    input[7:0]   opcode;
    output[7:0]  errst;
    reg[31:0]    data;
    reg[31:0]    data3;
begin
    errst = 0;

```

## Typical Application

```
data = 0;
data3 = 0;
while(!(data == edata) && !errst) // Wait operation
begin: BR_EN
RD16(addr, data); // Read status
if(data[7] != edata[7])
begin
if(data[5] == 1)
begin
RD16(addr, data3);
RD16(addr, data);
if(data[6] != data3[6])
begin
$display("CHECK_STATUS: Error timeout on single data program");
errst = 1;
disable BR_EN;
end
end
else
begin
if(opcode == 2)
if(data[1] == 1)
begin
RD16(addr, data3);
if(port_size == 32)
RD32(addr, data);
else
RD16(addr, data);
if(data[1] == 1 && data != edata)
begin
$display("CHECK_STATUS: Error on write buffer");
errst = 3;
disable BR_EN;
end
end
end
end
else
begin
RD16(addr, data3);
if(port_size == 32)
RD32(addr, data);
else
begin
RD16(addr, data);
edata[31:16] = 16'h0;
end
if(data != edata)
begin
$display("CHECK_STATUS: Error in data write on single data program");
errst = 2;
disable BR_EN;
end
end
end
endtask
```

## 20.7.8 8 bit support

This section details the pin connections for Intel mode and Motorola mode.

Intel Mode - For intel mode use the following connection:

**Table 20-10. Intel Mode pin connections**

ARM platform Pin	EIM Pin	Notes
ADS#	IPP_DO_ADV_B	WAL = 1, RAL = 1
W/R	IPP_DO_BE_B	WBED = 1
WR#	WE#	
RD#	OE#	

Mot. Mode - For intel mode use the following connection:

**Table 20-11. Motorola Mode pin connections**

ARM platform Pin	EIM Pin	Notes
AS#	IPP_DO_CS_B	
R/W#	WE#	
LDS#	BE#	

## 20.8 External Bus Timing Diagrams

The following timing diagrams show the timing of accesses to memory or a peripheral with different timing parameters. All examples done for CS0, but are valid for any others chip select. BE means one from current used BE[3:0].

## 20.8.1 Asynchronous Read Memory Accesses Timing Diagram

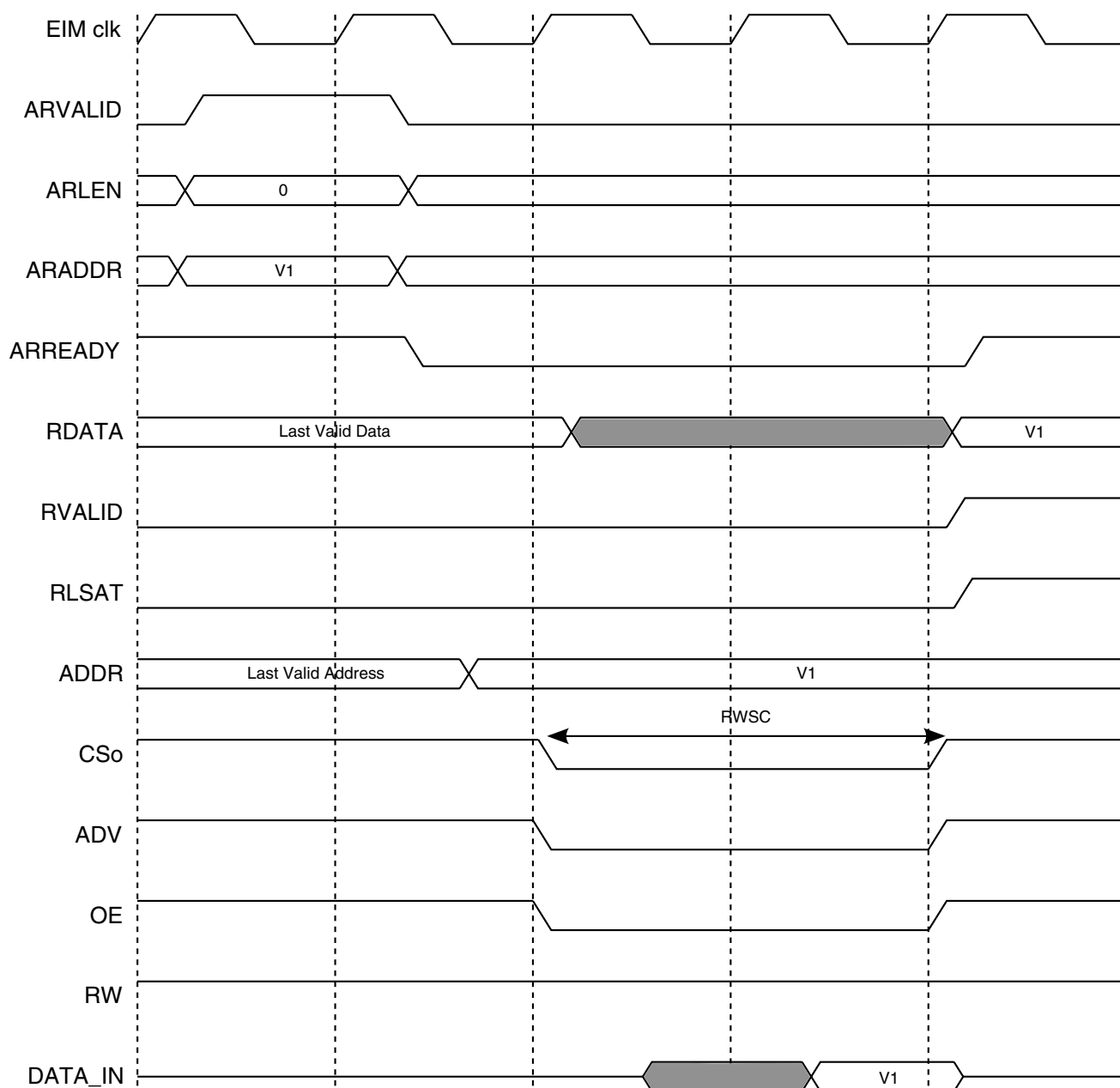
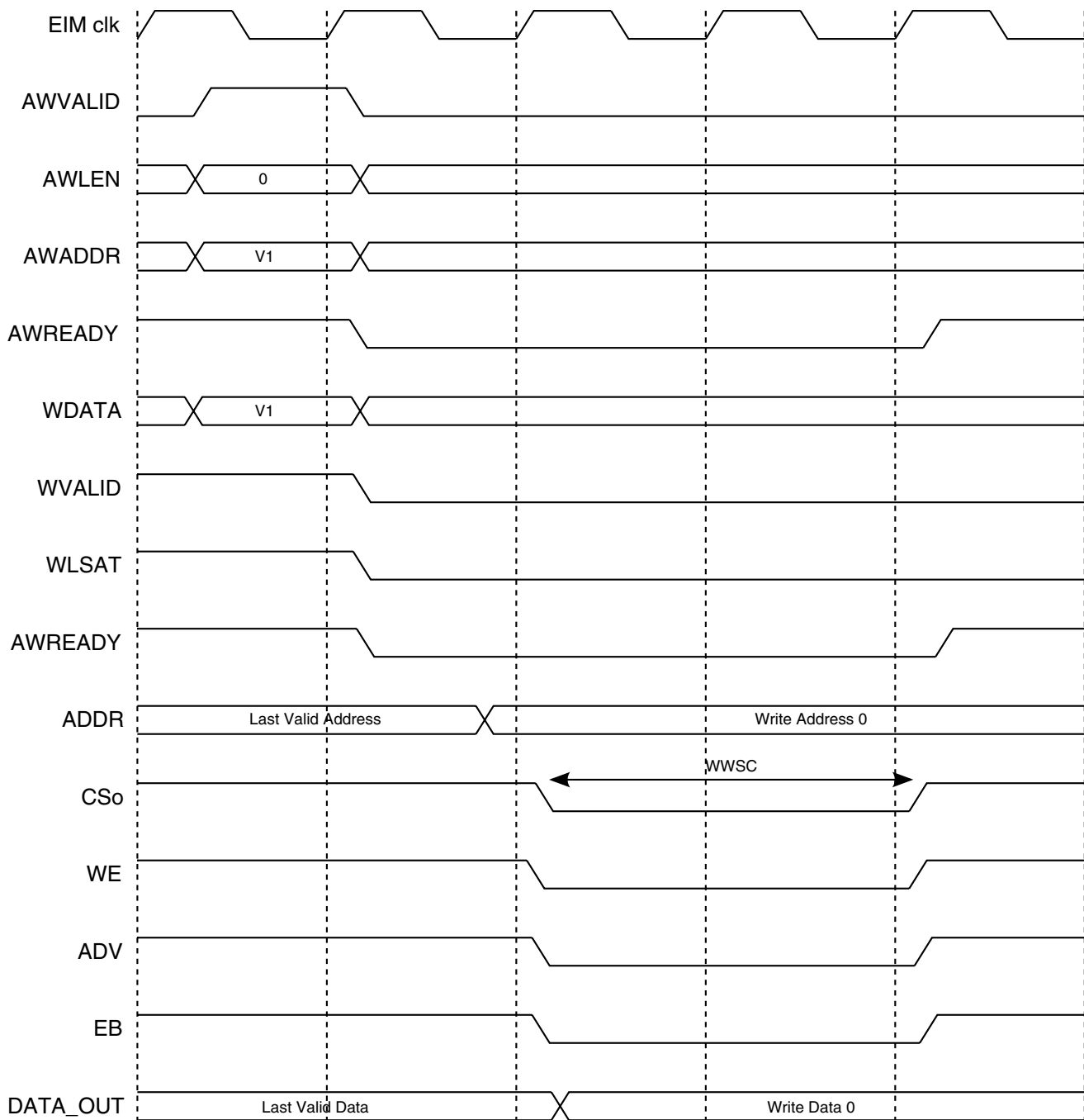


Figure 20-2. Read Access,  $RWSC=2, RCSA=0, OEA=0, RCSN=0, OEN=0, RAL=1$

## 20.8.2 Asynchronous Write Memory Accesses Timing Diagram



**Figure 20-3. Write Access, WWSC=2, WCSA=0, WEA=0, WCSN=0, WEN=0, BEA=0, BEN=0, WAL=1**

## 20.8.3 Asynchronous Read/Write Memory Accesses Timing Diagram

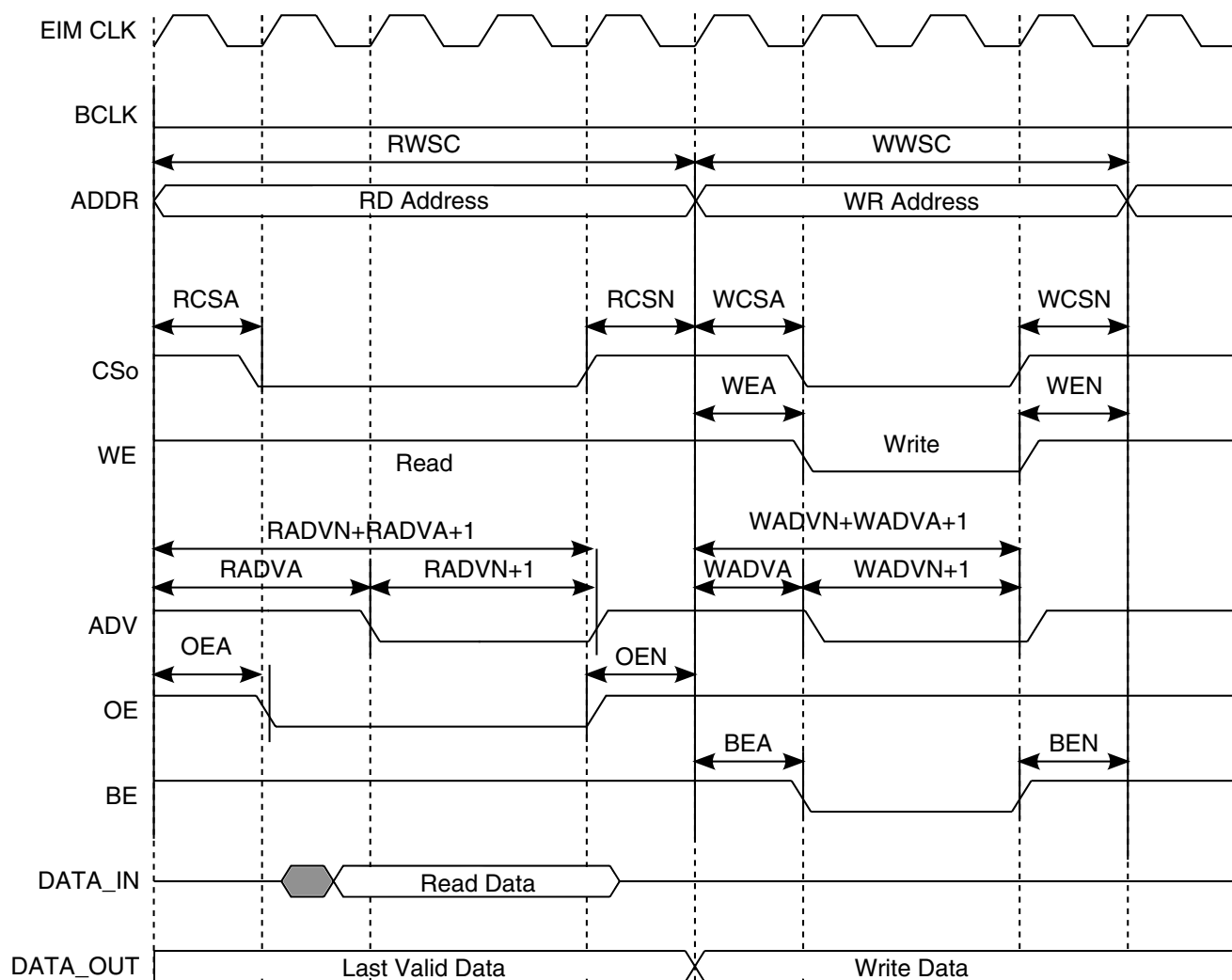


Figure 20-4.

$RCSA=1, RADVA=2, OEA=1, RADVN=1, RCSN=1, OEN=1, WCSA=1, WEA=1, WADVA=1, BEA=1, WADVN=1, WCSN=1, WEN=1, BEN=1$



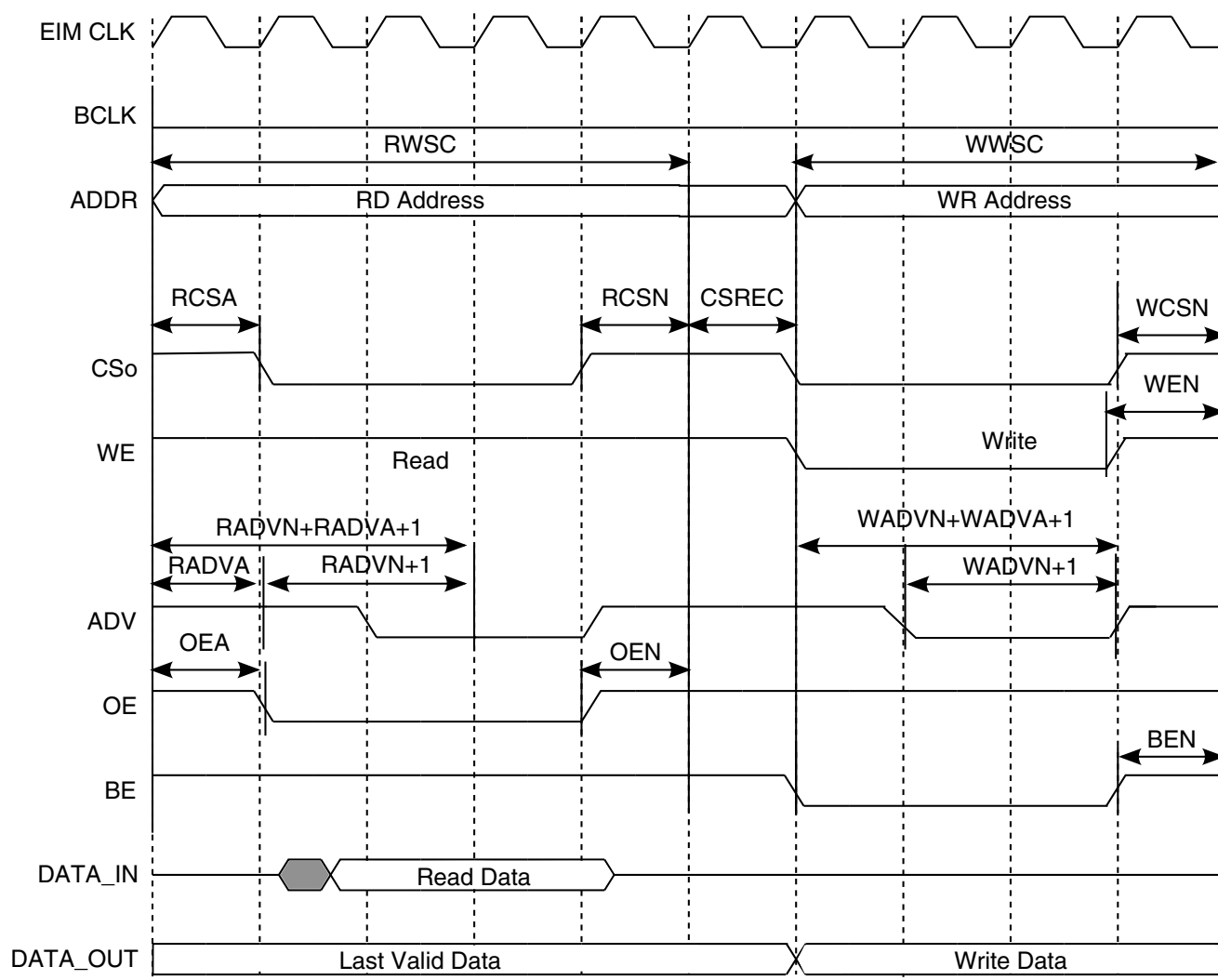


Figure 20-5.

**RWSC=5, RCSA=1, RCSN=1, RADVA=1, RADVN=1, OEA=1, OEN=1, WWSC=4, WCSA=0, WCSN=1, WEA=0, WEN=1, WADVA=1, WADVN=1, BEA=0, BEN=1, CSREC=1**

## 20.8.4 Asynchronous Read/Write Using RAL, WAL and CSREC

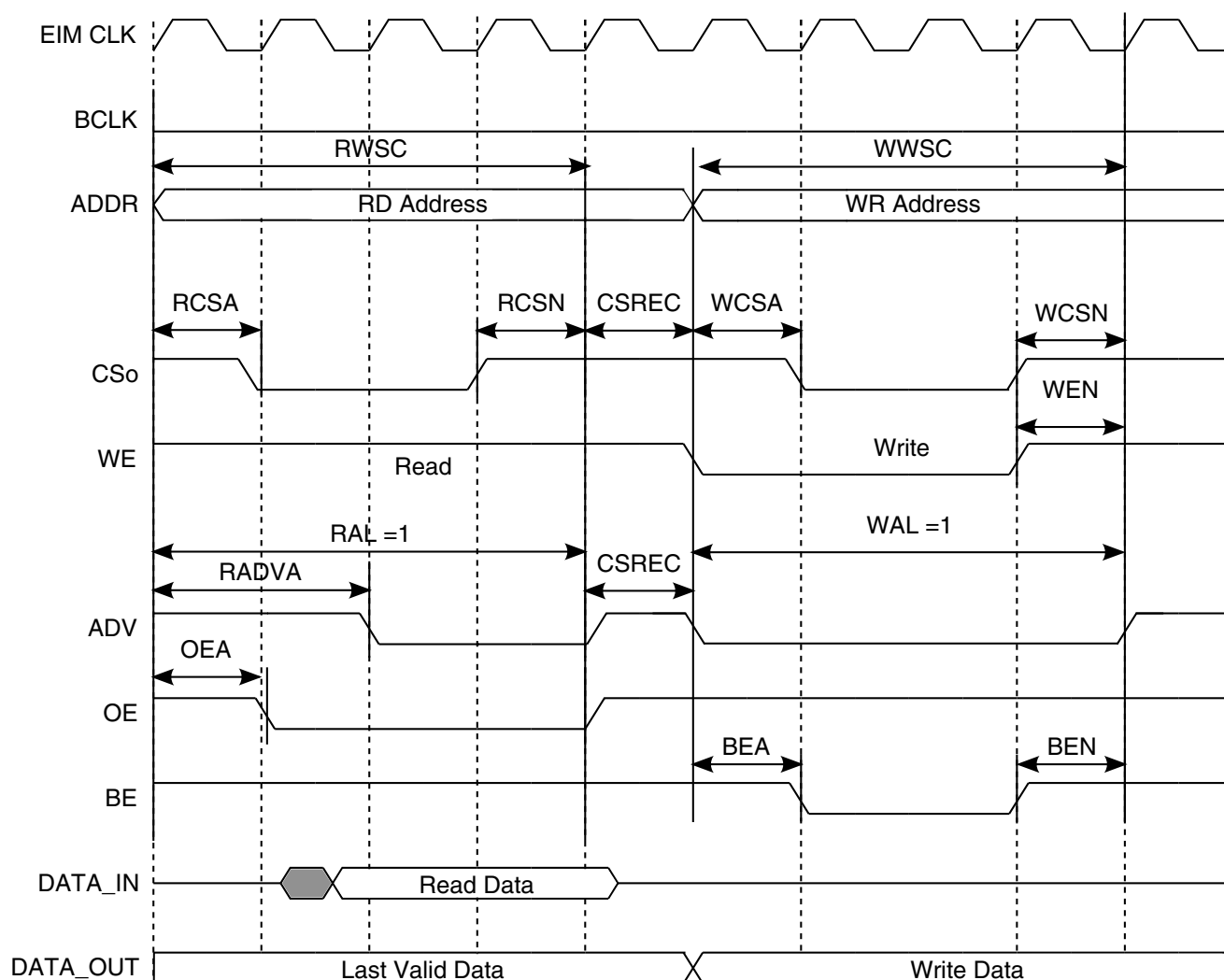


Figure 20-6.

**RAL=1,RCSN=1,RADVA=2,OEA=1,RCSN=1,CSREC=1,WCSA=1,WEA=0,WADVA=0,BEA=1,WAL=1,WCSN=1,WEN=1,BEN=1**

## 20.8.5 Consecutive Asynchronous Write Memory Accesses Timing Diagram

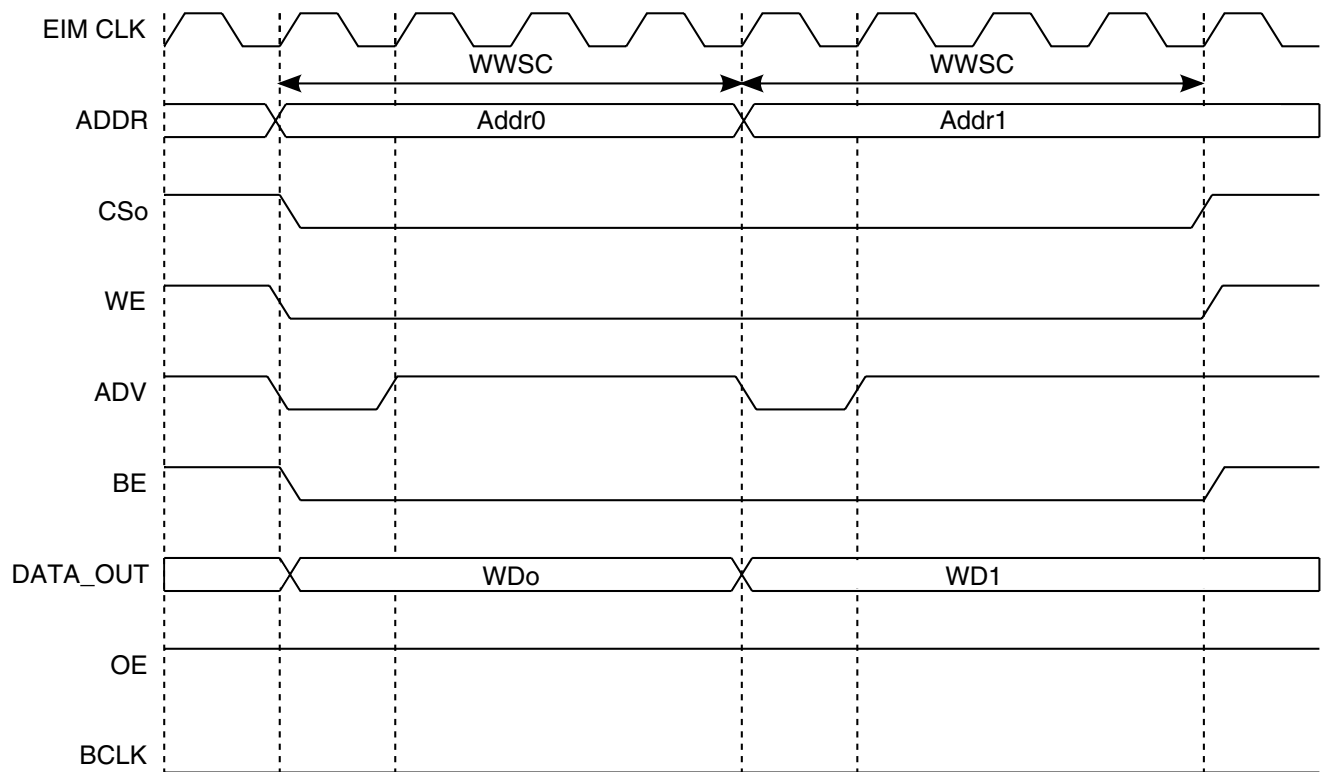
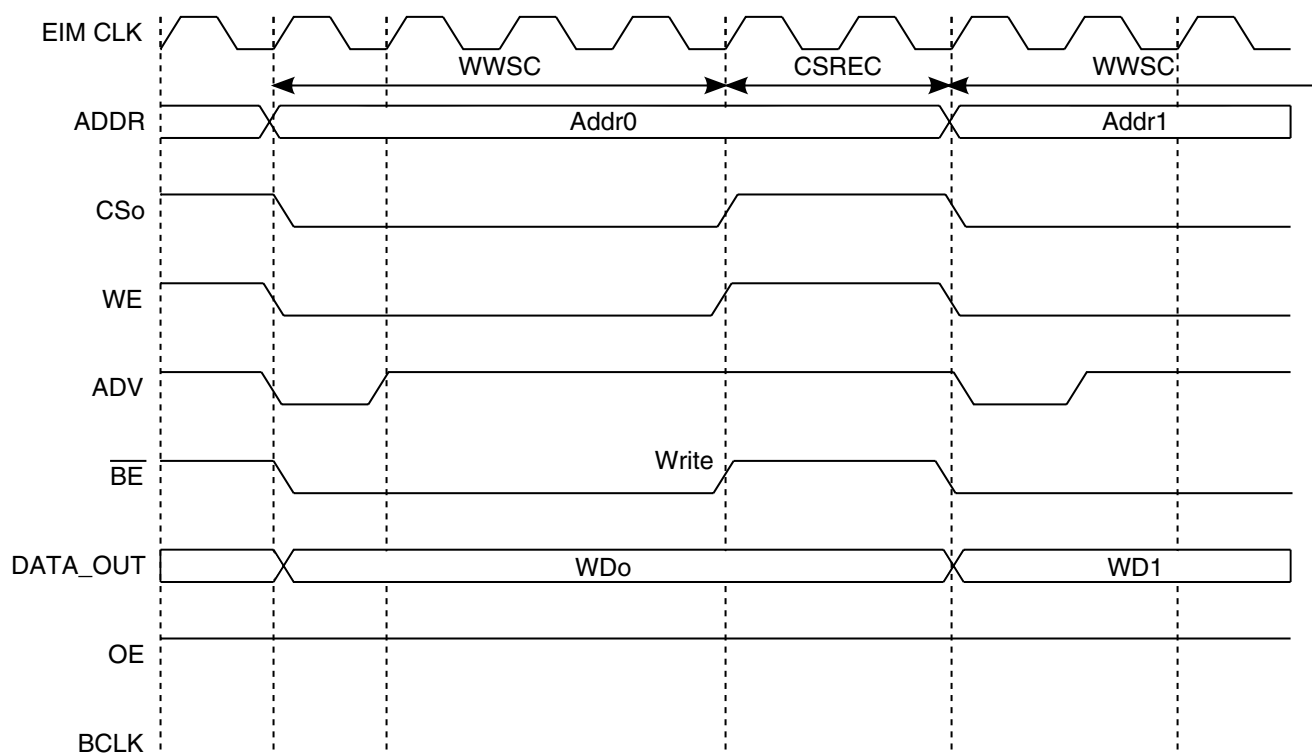


Figure 20-7.

WWSC=4, WCSA=0, WEA=0, WADVA=0, BEA=0, WCSN=0, WEN=0, WADV=0, BEN=0, CSRE  
C=0

## External Bus Timing Diagrams



**Figure 20-8.**

**WWSC=4, WCSA=0, WEA=0, WADVA=0, BEA=0, WCSN=0, WEN=0, WADV=0, BEN=0, CSRE C=2**

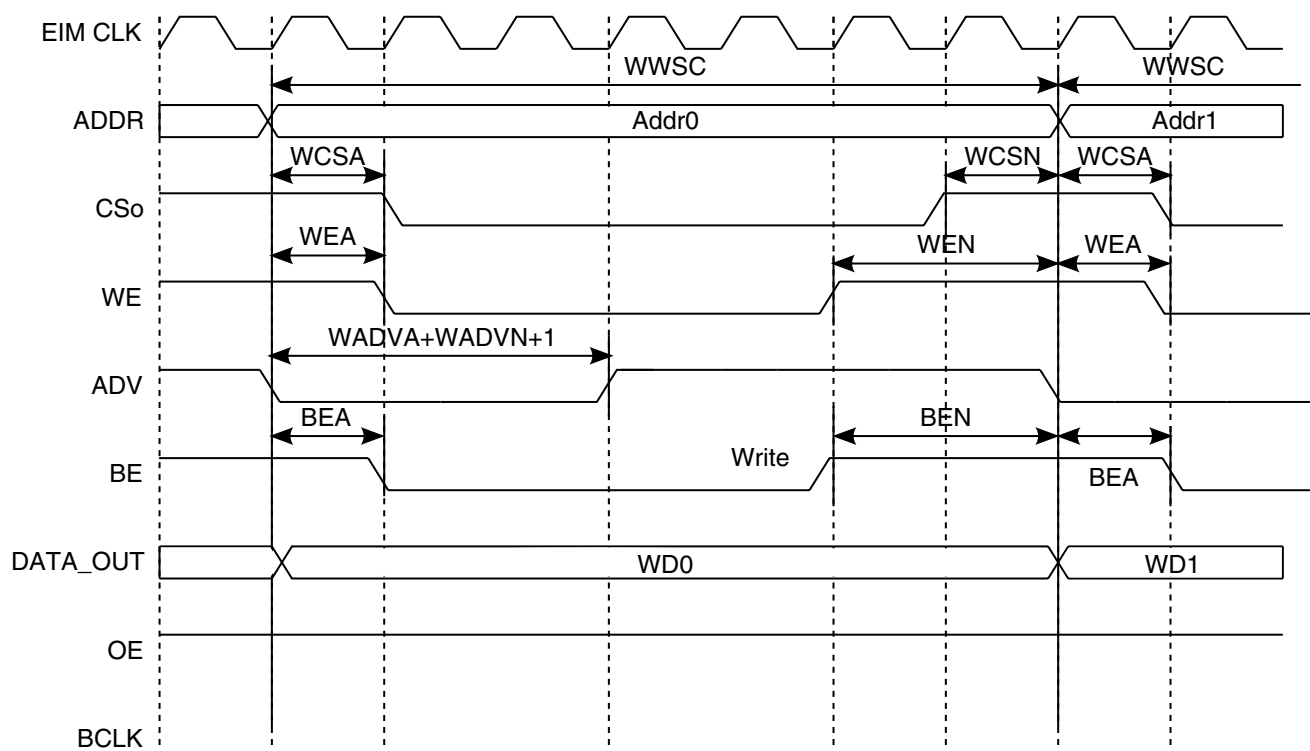
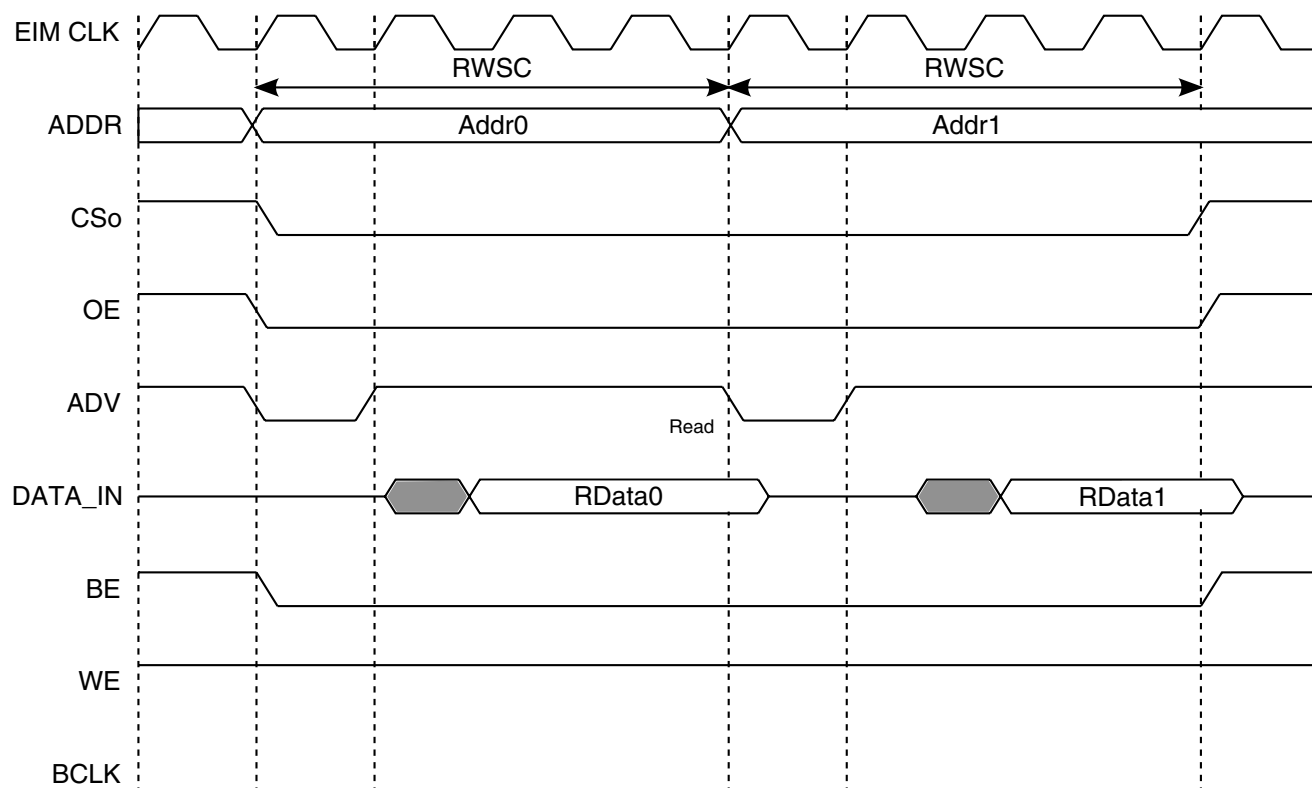


Figure 20-9.

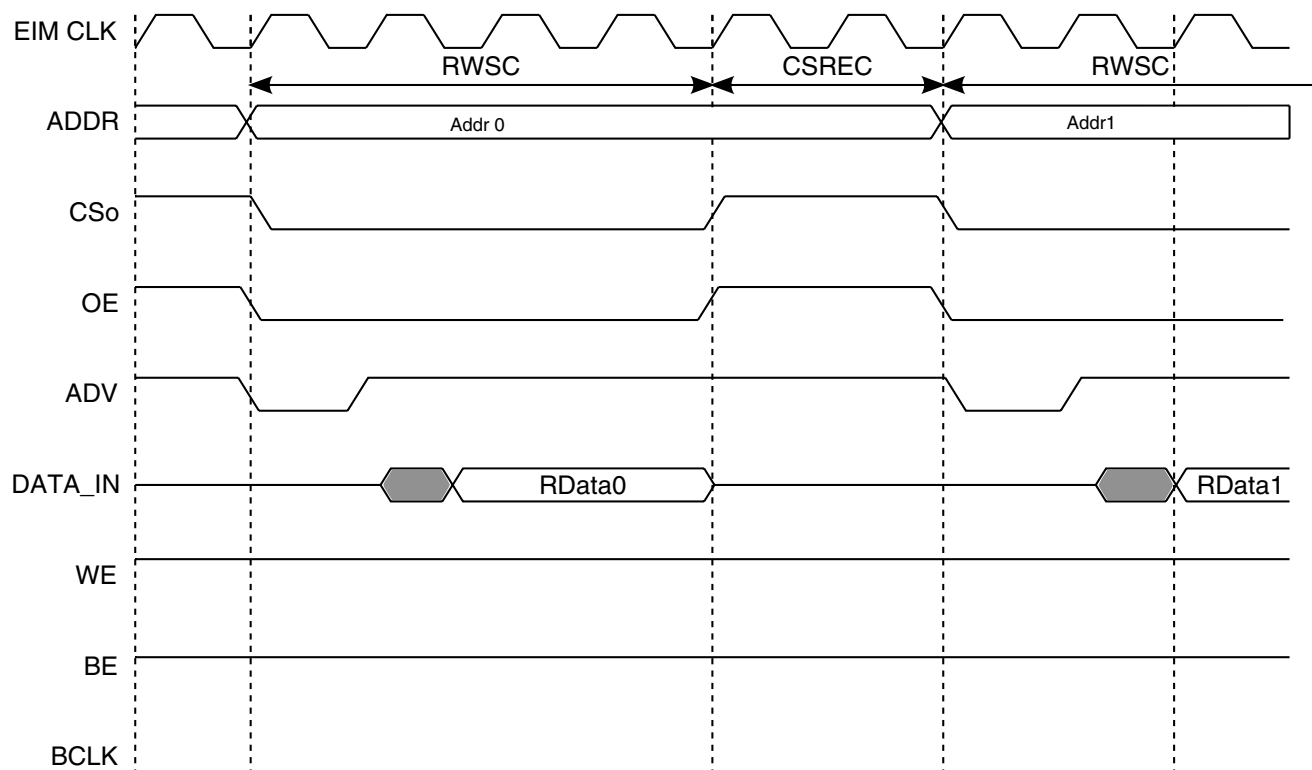
WWSC=7, WCSA=1, WCSN=1, WEA=1, WEN=2, WADVA=0, WADV N=2, BEA=1, BEN=2

## 20.8.6 Consecutive Asynchronous Read Memory Accesses Timing Diagram

## External Bus Timing Diagrams

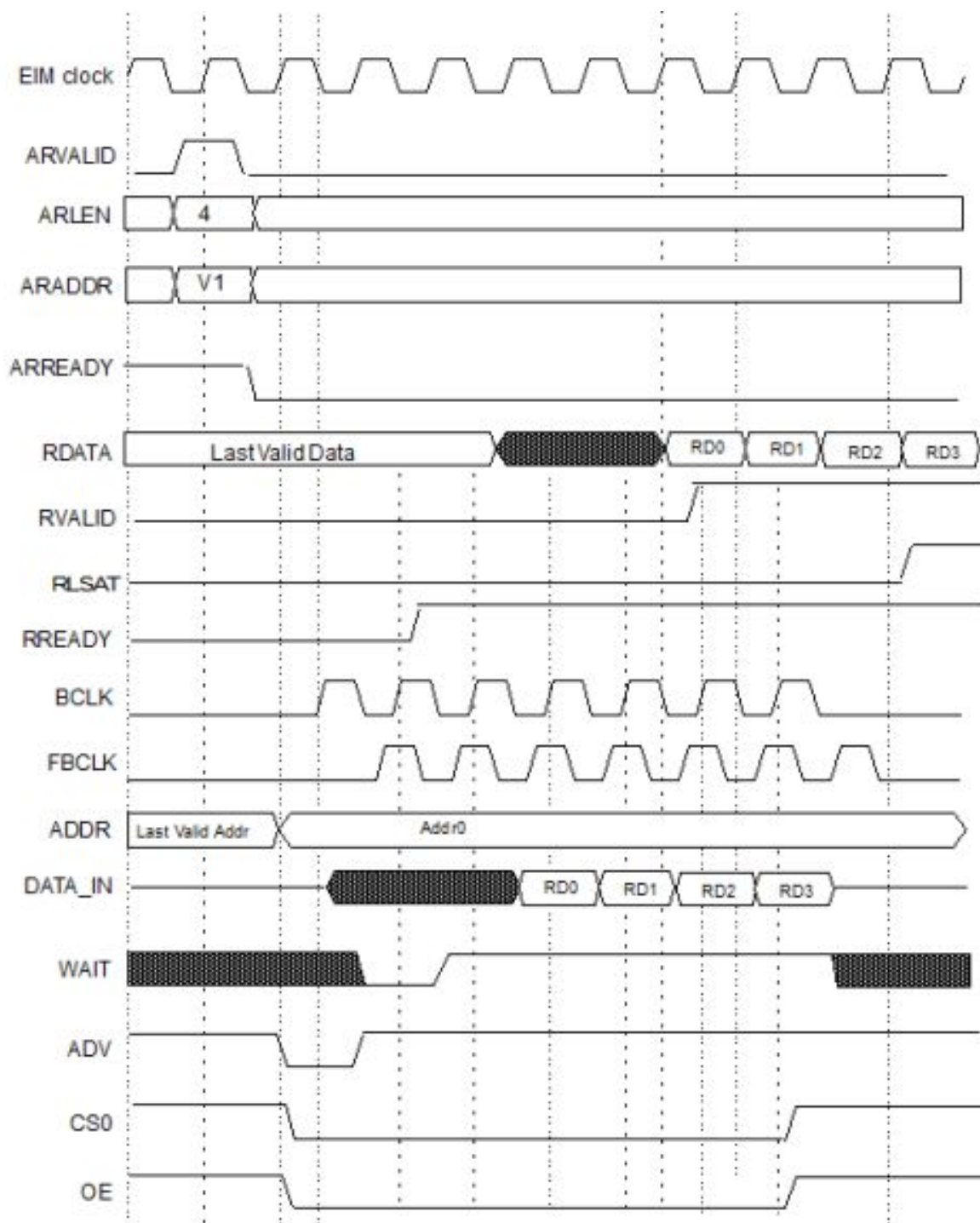


**Figure 20-10. RWSC=4, RCSA=0, OEA=0, RADVA=0, RCSN=0, OEN=0, RADVN=0, CSREC=0**



**Figure 20-11. RWSC=4, RCSA=0, OEA=0, RADVA=0, RCSN=0, OEN=0, RADVN=0, CSREC=2**

### 20.8.7 Burst (Synchronous Mode) Read Memory Accesses Timing Diagram - BCD=0



**Figure 20-12. SRD=1,BCD=0,BCS=0,RWSC=1,RADVA=0,RADV N=0,RFL=0,RL=0**

## 20.8.8 Burst (Synchronous Mode) Read Memory Accesses Timing Diagram - BCD=1

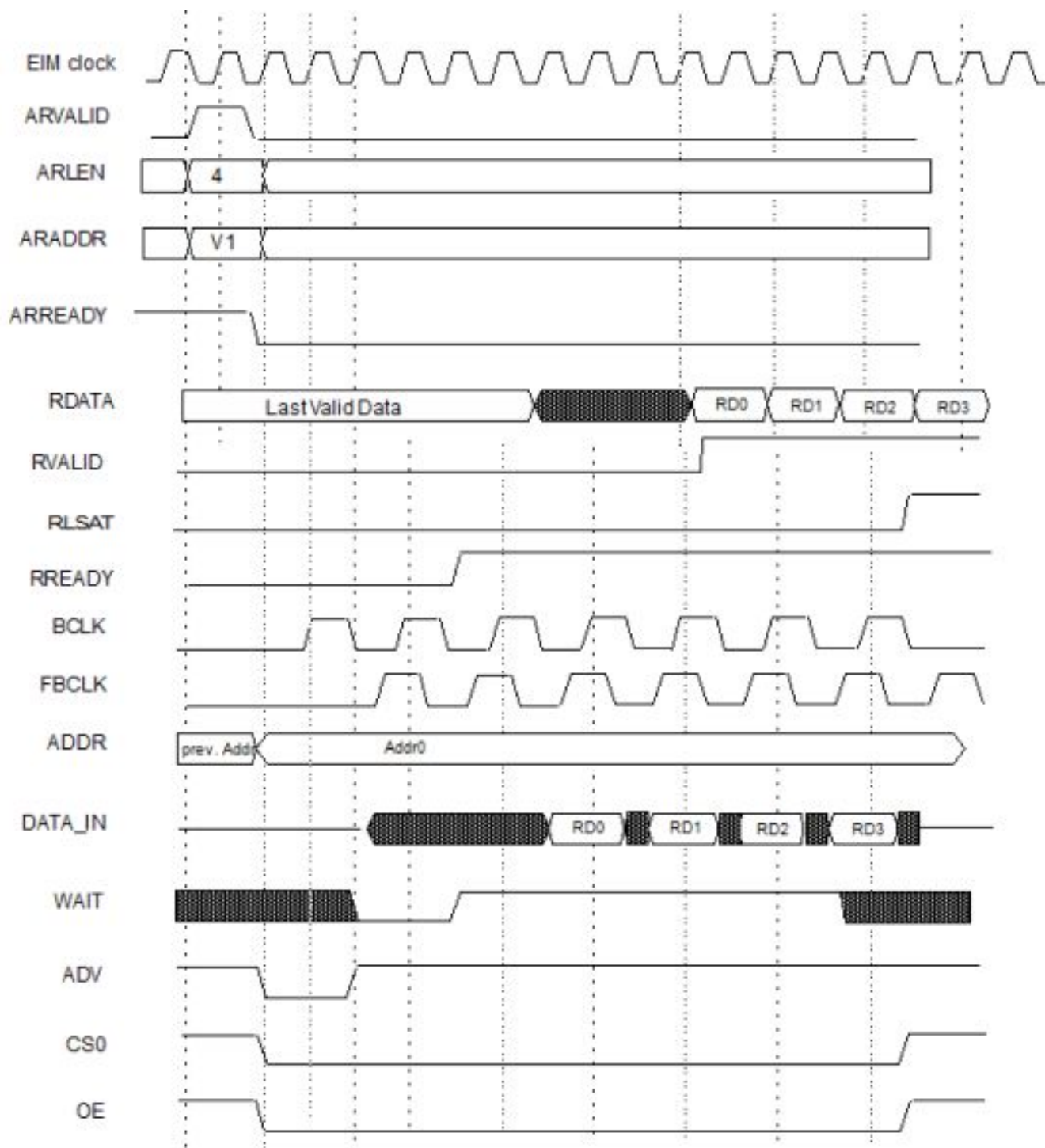


Figure 20-13. BCD=1, RL = 3



### 20.8.9 Asynchronous Page Mode Access

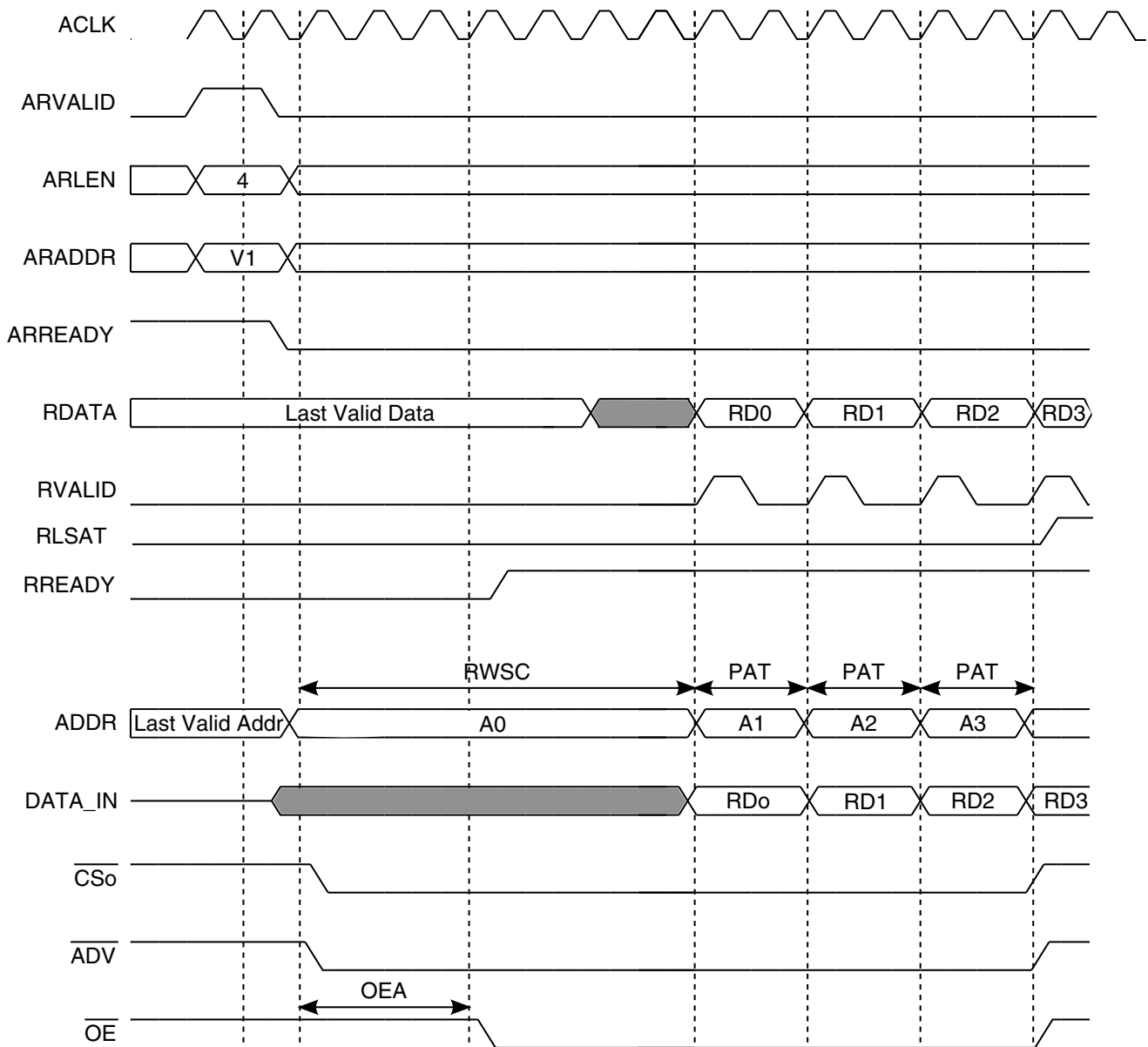
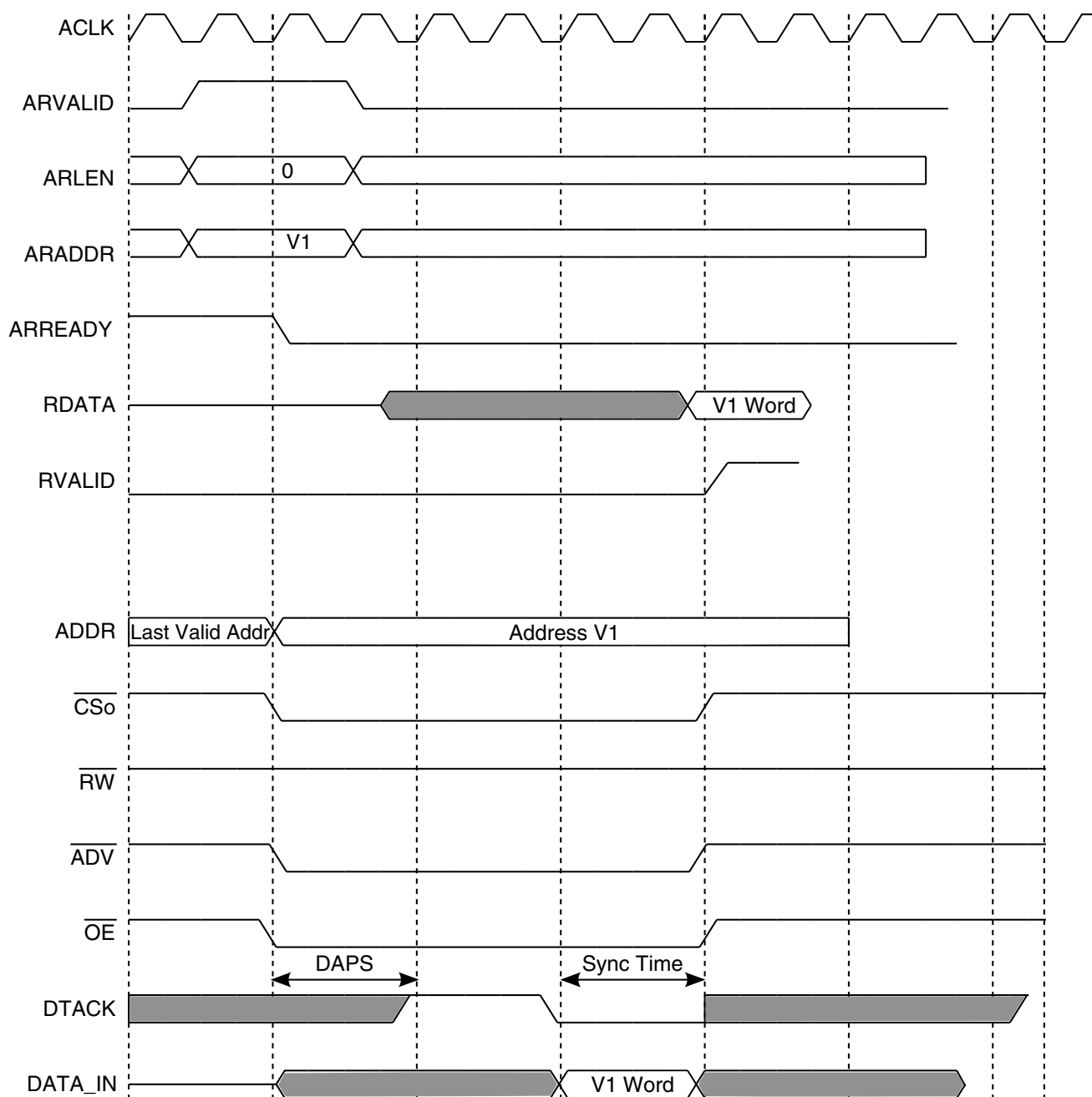


Figure 20-14. PAT = 2

### 20.8.10 DTACK Mode - AXI Single Access

## External Bus Timing Diagrams



**Figure 20-15. DAPS = 2**

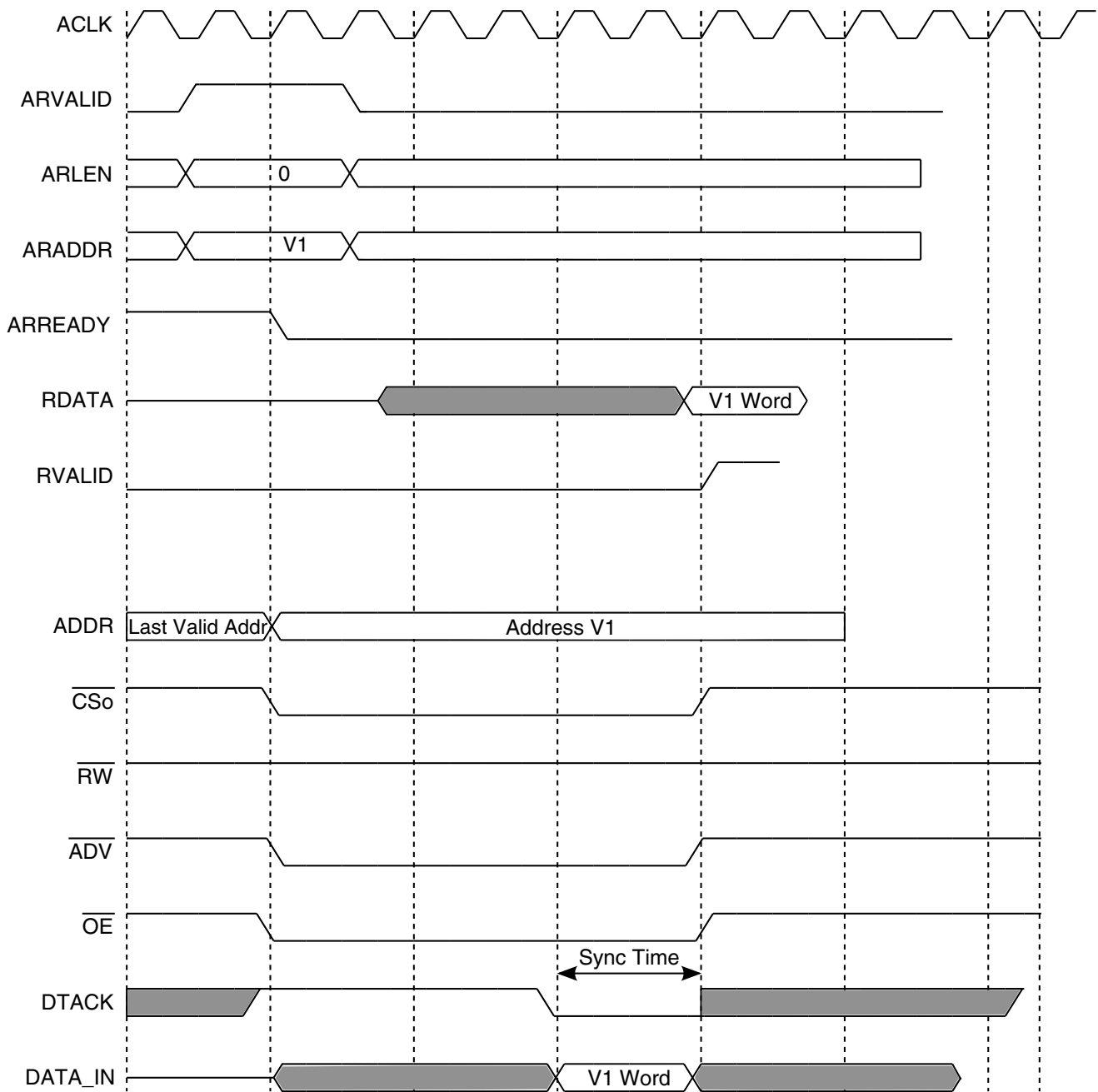
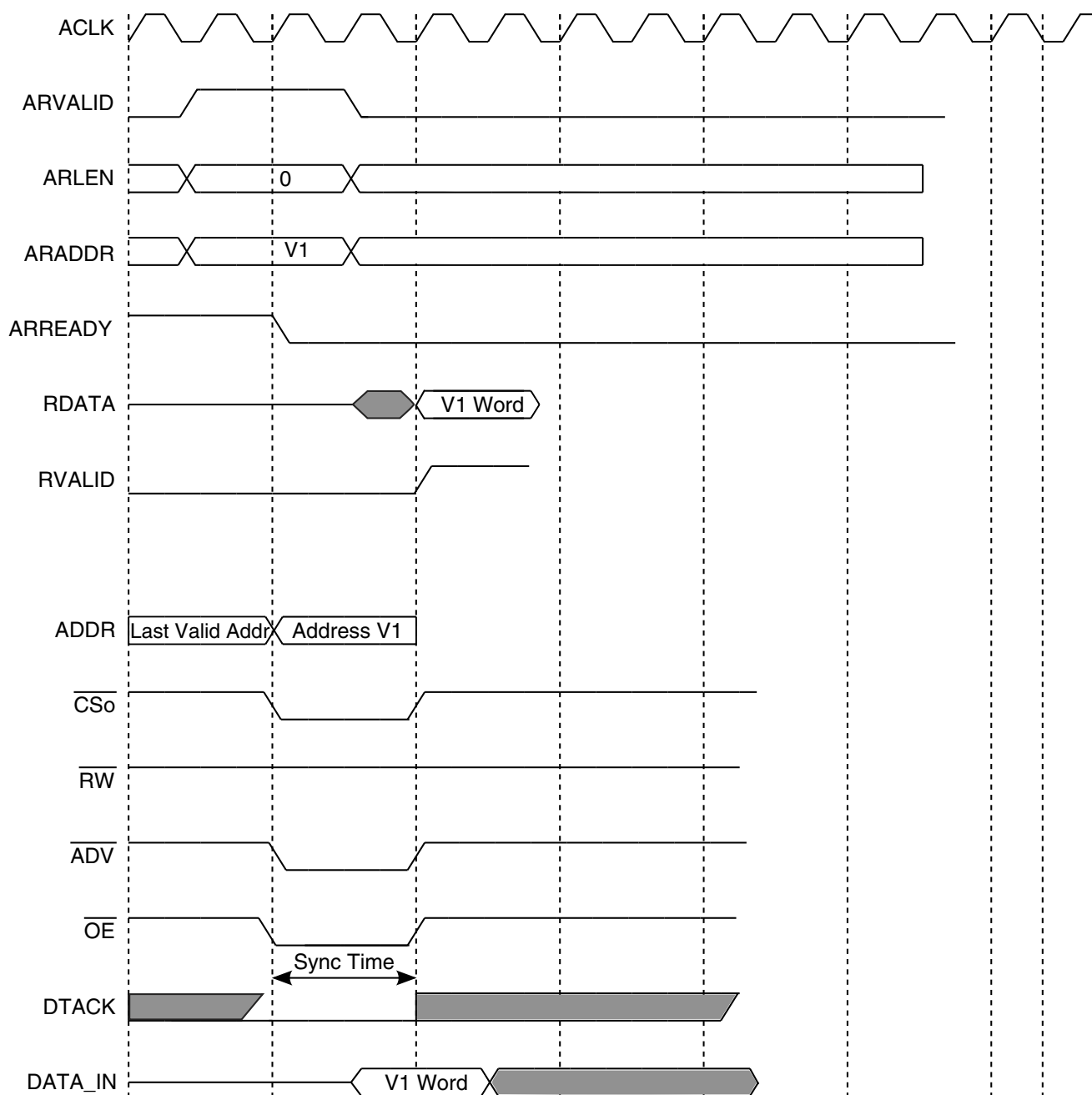
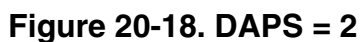


Figure 20-16. DAPS = 0

## External Bus Timing Diagrams



**Figure 20-17. DAPS = 0**



## 20.8.12 DTACK Mode - AXI Burst Access

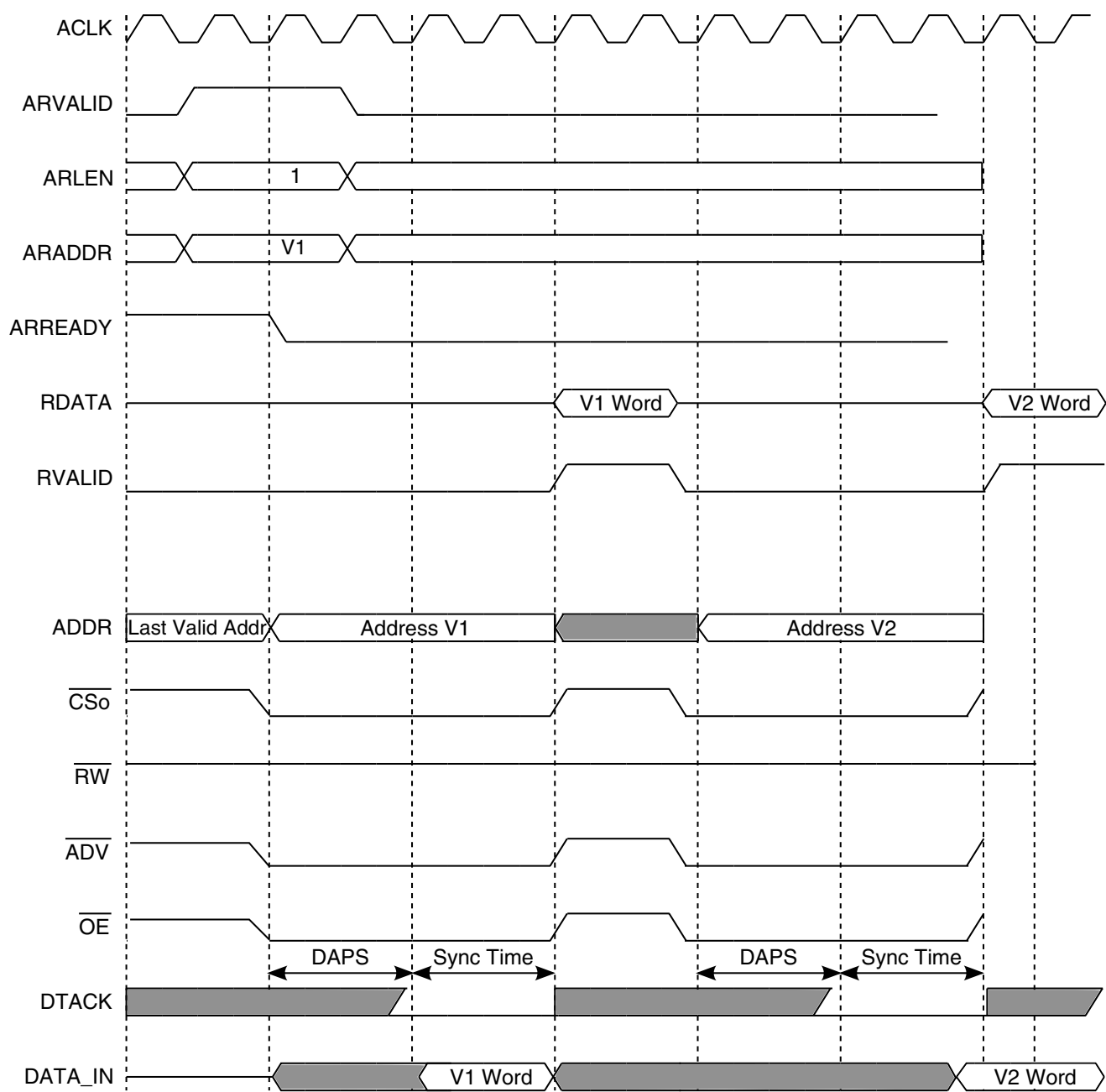


Figure 20-19. DAPS = 2 CSREC = 2

## 20.9 EIM Memory Map/Register Definition

The EIM includes 33 user-accessible 32-bit registers. The the EIM Configuration Register (EIM\_WCR) contains control bits that configure the EIM for certain operation modes.

The 160 bits used to control Individual Chip Select are divided into five registers:

- Chip Select n General Configuration Register 1 (EIM\_CSnGCR1)
- Chip Select n General Configuration Register 2 (EIM\_CSnGCR2)
- Chip Select n Read Configuration Register 1 (EIM\_CSnRCR1)
- Chip Select n Read Configuration Register 2 (EIM\_CSnRCR2)
- Chip Select n Write Configuration Register (EIM\_CSnWCR)

In addition there are 3 general registers: EIM\_WCR, EIM\_WIAR & EIM\_EAR.

### NOTE

- All EIM registers are sampled by IPG\_CLK\_S, therefore IPG\_CLK\_S must be active when accessing through IP bus.
- Read access from all registers (except EIM\_WIAR & EIM\_EAR) will generate one IPG\_XFR\_WAIT cycle.
- Read access from EIM\_WIAR & EIM\_EAR will generate six IPG\_XFR\_WAIT cycles.
- Write access to all registers (except EIM\_EAR) will generate three IPG\_XFR\_WAIT cycles.
- Write access to EIM\_EAR will generate six IPG\_XFR\_WAIT cycles.

### EIM memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
21B_8000	Chip Select n General Configuration Register 1 (EIM_CS0GCR1)	32	R/W	0001_0080h	<a href="#">20.9.1/813</a>
21B_8004	Chip Select n General Configuration Register 2 (EIM_CS0GCR2)	32	R/W	0000_1000h	<a href="#">20.9.2/818</a>
21B_8008	Chip Select n Read Configuration Register 1 (EIM_CS0RCR1)	32	R/W	0000_0000h	<a href="#">20.9.3/819</a>
21B_800C	Chip Select n Read Configuration Register 2 (EIM_CS0RCR2)	32	R/W	0000_0000h	<a href="#">20.9.4/822</a>
21B_8010	Chip Select n Write Configuration Register 1 (EIM_CS0WCR1)	32	R/W	0000_0000h	<a href="#">20.9.5/823</a>
21B_8014	Chip Select n Write Configuration Register 2 (EIM_CS0WCR2)	32	R/W	0000_0000h	<a href="#">20.9.6/826</a>
21B_8018	Chip Select n General Configuration Register 1 (EIM_CS1GCR1)	32	R/W	0001_0080h	<a href="#">20.9.1/813</a>

*Table continues on the next page...*

## EIM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
21B_801C	Chip Select n General Configuration Register 2 (EIM_CS1GCR2)	32	R/W	0000_1000h	<a href="#">20.9.2/818</a>
21B_8020	Chip Select n Read Configuration Register 1 (EIM_CS1RCR1)	32	R/W	0000_0000h	<a href="#">20.9.3/819</a>
21B_8024	Chip Select n Read Configuration Register 2 (EIM_CS1RCR2)	32	R/W	0000_0000h	<a href="#">20.9.4/822</a>
21B_8028	Chip Select n Write Configuration Register 1 (EIM_CS1WCR1)	32	R/W	0000_0000h	<a href="#">20.9.5/823</a>
21B_802C	Chip Select n Write Configuration Register 2 (EIM_CS1WCR2)	32	R/W	0000_0000h	<a href="#">20.9.6/826</a>
21B_8030	Chip Select n General Configuration Register 1 (EIM_CS2GCR1)	32	R/W	0001_0080h	<a href="#">20.9.1/813</a>
21B_8034	Chip Select n General Configuration Register 2 (EIM_CS2GCR2)	32	R/W	0000_1000h	<a href="#">20.9.2/818</a>
21B_8038	Chip Select n Read Configuration Register 1 (EIM_CS2RCR1)	32	R/W	0000_0000h	<a href="#">20.9.3/819</a>
21B_803C	Chip Select n Read Configuration Register 2 (EIM_CS2RCR2)	32	R/W	0000_0000h	<a href="#">20.9.4/822</a>
21B_8040	Chip Select n Write Configuration Register 1 (EIM_CS2WCR1)	32	R/W	0000_0000h	<a href="#">20.9.5/823</a>
21B_8044	Chip Select n Write Configuration Register 2 (EIM_CS2WCR2)	32	R/W	0000_0000h	<a href="#">20.9.6/826</a>
21B_8048	Chip Select n General Configuration Register 1 (EIM_CS3GCR1)	32	R/W	0001_0080h	<a href="#">20.9.1/813</a>
21B_804C	Chip Select n General Configuration Register 2 (EIM_CS3GCR2)	32	R/W	0000_1000h	<a href="#">20.9.2/818</a>
21B_8050	Chip Select n Read Configuration Register 1 (EIM_CS3RCR1)	32	R/W	0000_0000h	<a href="#">20.9.3/819</a>
21B_8054	Chip Select n Read Configuration Register 2 (EIM_CS3RCR2)	32	R/W	0000_0000h	<a href="#">20.9.4/822</a>
21B_8058	Chip Select n Write Configuration Register 1 (EIM_CS3WCR1)	32	R/W	0000_0000h	<a href="#">20.9.5/823</a>
21B_805C	Chip Select n Write Configuration Register 2 (EIM_CS3WCR2)	32	R/W	0000_0000h	<a href="#">20.9.6/826</a>
21B_8060	Chip Select n General Configuration Register 1 (EIM_CS4GCR1)	32	R/W	0001_0080h	<a href="#">20.9.1/813</a>
21B_8064	Chip Select n General Configuration Register 2 (EIM_CS4GCR2)	32	R/W	0000_1000h	<a href="#">20.9.2/818</a>
21B_8068	Chip Select n Read Configuration Register 1 (EIM_CS4RCR1)	32	R/W	0000_0000h	<a href="#">20.9.3/819</a>
21B_806C	Chip Select n Read Configuration Register 2 (EIM_CS4RCR2)	32	R/W	0000_0000h	<a href="#">20.9.4/822</a>
21B_8070	Chip Select n Write Configuration Register 1 (EIM_CS4WCR1)	32	R/W	0000_0000h	<a href="#">20.9.5/823</a>

Table continues on the next page...



**EIM memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
21B_8074	Chip Select n Write Configuration Register 2 (EIM_CS4WCR2)	32	R/W	0000_0000h	<a href="#">20.9.6/826</a>
21B_8078	Chip Select n General Configuration Register 1 (EIM_CS5GCR1)	32	R/W	0001_0080h	<a href="#">20.9.1/813</a>
21B_807C	Chip Select n General Configuration Register 2 (EIM_CS5GCR2)	32	R/W	0000_1000h	<a href="#">20.9.2/818</a>
21B_8080	Chip Select n Read Configuration Register 1 (EIM_CS5RCR1)	32	R/W	0000_0000h	<a href="#">20.9.3/819</a>
21B_8084	Chip Select n Read Configuration Register 2 (EIM_CS5RCR2)	32	R/W	0000_0000h	<a href="#">20.9.4/822</a>
21B_8088	Chip Select n Write Configuration Register 1 (EIM_CS5WCR1)	32	R/W	0000_0000h	<a href="#">20.9.5/823</a>
21B_808C	Chip Select n Write Configuration Register 2 (EIM_CS5WCR2)	32	R/W	0000_0000h	<a href="#">20.9.6/826</a>
21B_8090	EIM Configuration Register (EIM_WCR)	32	R/W	<a href="#">See section</a>	<a href="#">20.9.7/827</a>
21B_8094	DLL Control Register (EIM_DCR)	32	R/W	0014_0000h	<a href="#">20.9.8/829</a>
21B_8098	DLL Status Register (EIM_DSR)	32	R	0000_0000h	<a href="#">20.9.9/830</a>
21B_809C	EIM IP Access Register (EIM_WIAR)	32	R/W	0000_0010h	<a href="#">20.9.10/831</a>
21B_80A0	Error Address Register (EIM_EAR)	32	R/W	0000_0000h	<a href="#">20.9.11/832</a>

## 20.9.1 Chip Select n General Configuration Register 1 (EIM\_CS<sub>n</sub>GCR1)

Address: 21B\_8000h base + 0h offset + (24d × i), where i=0d to 5d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
R	PSZ				WP	GBC				AUS	CSREC			SP	DSZ			
W																		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1		
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
R	BCS			BCD			WC	BL			CREP	CRE	RFL	WFL	MUM	SRD	SWR	CSEN
W																		
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	

EIM\_CS<sub>n</sub>GCR1 field descriptions

Field	Description
31–28 PSZ	<p>Page Size. This bit field indicates memory page size in words (word is defined by the DSZ field). PSZ is used when fix latency mode is applied, WFL=1 for sync. write accesses, RFL=1 for sync. Read accesses. When working in fix latency mode WAIT signal from the external device is not being monitored, PSZ is used to determine if page boundary is reached and renewal of access is preformed. This bit field is ignored when sync. Mode is disabled or fix latency mode is not being used for write or read access separately.</p> <p>It can be valid for both access type, read or write, or only for one type, according to configuration. PSZ is cleared by a hardware reset.</p> <p>0000 8 words page size  0001 16 words page size  0010 32 words page size  0011 64 words page size  0100 128 words page size  0101 256 words page size  0110 512 words page size  0111 1024 (1k) words page size  1000 2048 (2k) words page size  1001 - 1111 Reserved</p>
27 WP	<p>Write Protect. This bit prevents writes to the address range defined by the corresponding chip select. WP is cleared by a hardware reset.</p> <p>0 Writes are allowed in the memory range defined by chip.  1 Writes are prohibited. All attempts to write to an address mapped by this chip select result in a error response and no assertion of the chip select output.</p>
26–24 GBC	<p>Gap Between Chip Selects. This bit field, according to the settings shown below, determines the minimum time between end of access to the current chip select and start of access to different chip select. GBC is cleared by a hardware reset.</p> <p>Example settings:</p> <p>000 minimum of 0 EIM clock cycles before next access from different chip select (async. mode only)  001 minimum of 1 EIM clock cycles before next access from different chip select  010 minimum of 2 EIM clock cycles before next access from different chip select  111 minimum of 7 EIM clock cycles before next access from different chip select</p>
23 AUS	<p>Address UnShifted. This bit indicates an unshifted mode for address assertion for the relevant chip select accesses. AUS bit is cleared by hardware reset.</p> <p>0 Address shifted according to port size (DSZ config.)  1 Address unshifted</p>
22–20 CSREC	<p>CS Recovery. This bit field, according to the settings shown below, determines the minimum pulse width of CS, OE, and WE control signals before executing a new back to back access to the same chip select. CSREC is cleared by a hardware reset.</p> <p><b>NOTE:</b> The reset value for EIM_CS0GCR1, CSREC[2:0] is 0b110. For EIM_CS1GCR1 - EIM_CS5GCR, the reset value is 0b000.</p> <p>Example settings:</p> <p>000 0 EIM clock cycles minimum width of CS, OE and WE signals (read async. mode only)  001 1 EIM clock cycles minimum width of CS, OE and WE signals</p>

*Table continues on the next page...*

**EIM\_CSnGCR1 field descriptions (continued)**

Field	Description
	010 2 EIM clock cycles minimum width of CS, OE and WE signals 111 7 EIM clock cycles minimum width of CS, OE and WE signals
19 SP	Supervisor Protect. This bit prevents accesses to the address range defined by the corresponding chip select when the access is attempted in the User mode. SP is cleared by a hardware reset.  0 User mode accesses are allowed in the memory range defined by chip select. 1 User mode accesses are prohibited. All attempts to access an address mapped by this chip select in User mode results in an error response and no assertion of the chip select output.
18–16 DSZ	Data Port Size. This bit field defines the width of an external device's data port as shown below.  <b>NOTE:</b> Only async. access supported for 8 bit port.  <b>NOTE:</b> The reset value for EIM_CS0GCR1, DSZ[2] = 0, DSZ[1:0] = EIM_BOOT[1:0]. For EIM_CS1GCR1 - EIM_CS5GCR1, the reset value is 0b001.  000 Reserved. 001 16 bit port resides on DATA[15:0] 010 16 bit port resides on DATA[31:16] 011 32 bit port resides on DATA[31:0] 100 8 bit port resides on DATA[7:0] 101 8 bit port resides on DATA[15:8] 110 8 bit port resides on DATA[23:16] 111 8 bit port resides on DATA[31:24]
15–14 BCS	Burst Clock Start. When SRD=1 or SWR=1, this bit field determines the number of EIM clock cycles delay from start of access before the first rising edge of BCLK is generated.  When BCD=0 value of BCS=0 results in a half clock delay after the start of access. For other values of BCD a one clock delay after the start of access is applied, not an immediate assertion. BCS is cleared by a hardware reset.  00 0 EIM clock cycle additional delay 01 1 EIM clock cycle additional delay 10 2 EIM clock cycle additional delay 11 3 EIM clock cycle additional delay
13–12 BCD	Burst Clock Divisor. This bit field contains the value used to program the burst clock divisor for BCLK generation. It is used to divide the internal EIMbus frequency. BCD is cleared by a hardware reset.  <b>NOTE:</b> For other than the mentioned below frequency such as 104 MHz, EIM clock (input clock) should be adjust accordingly.  00 Divide EIM clock by 1 01 Divide EIM clock by 2 10 Divide EIM clock by 3 11 Divide EIM clock by 4
11 WC	Write Continuous. The WI bit indicates that write access to the memory are always continuous accesses regardless of the BL field value. WI is cleared by hardware reset.  0 Write access burst length occurs according to BL value. 1 Write access burst length is continuous.
10–8 BL	Burst Length. The BL bit field indicates memory burst length in words (word is defined by the DSZ field) and should be properly initialized for mixed wrap/increment accesses support. Continuous BL value corresponds to continuous burst length setting of the external memory device. For fix memory burst size,

*Table continues on the next page...*

**EIM\_CS $n$ GCR1 field descriptions (continued)**

Field	Description
	<p>type is always wrap. In case not matching wrap boundaries in both the memory (BL field) and Master access on the current address, EIM update address on the external device address bus and regenerates the access.</p> <p>BL is cleared by a hardware reset.</p> <p>When APR=1, Page Read Mode is applied, BL determine the number of words within the read page burst. BL is cleared by a hardware reset for EIM_CS0GCR1 - EIM_CS5GCR1.</p> <p>000 4 words Memory wrap burst length (read page burst size when APR = 1)</p> <p>001 8 words Memory wrap burst length (read page burst size when APR = 1)</p> <p>010 16 words Memory wrap burst length (read page burst size when APR = 1)</p> <p>011 32 words Memory wrap burst length (read page burst size when APR = 1)</p> <p>100 Continuous burst length (2 words read page burst size when APR = 1)</p> <p>101 Reserved</p> <p>110 Reserved</p> <p>111 Reserved</p>
7 CREP	<p>Configuration Register Enable Polarity. This bit indicates CRE memory pin assertion state, active-low or active-high, while executing a memory register set command to the external device (PSRAM memory type). CREP is set by a hardware reset.</p> <p><b>NOTE:</b> Whenever PSRAM is connected the CREP value must be correct also for accesses where CRE is disabled.</p> <p>For Non-PSRAM memory CREP value should be 1.</p> <p>0 CRE signal is active low</p> <p>1 CRE signal is active high</p>
6 CRE	<p>Configuration Register Enable. This bit indicates CRE memory pin state while executing a memory register set command to PSRAM external device. CRE is cleared by a hardware reset.</p> <p>0 CRE signal use is disable</p> <p>1 CRE signal use is enable</p>
5 RFL	<p>Read Fix Latency. This bit field determine if the controller is monitoring the WAIT signal from the External device connected to the chip select (handshake mode - fix or variable data latency) or if it start sampling data according to RWSC field, it only valid in synchronous mode. RFL is cleared by a hardware reset.</p> <p>When RFL=1 Burst access is terminated on page boundary and resume on the following page according to BL bit field configuration, because WAIT signal is not monitored from the external device.</p> <p>0 the External device WAIT signal is being monitored, and it reflect the external data bus state</p> <p>1 the state of the External devices is determined internally (Fix latency mode only)</p>
4 WFL	<p>Write Fix Latency. This bit field determine if the controller is monitoring the WAIT signal from the External device connected to the chip select (handshake mode - fix or variable data latency) or if it start data transfer according to WWSC field, it only valid in synchronous mode. WFL is cleared by a hardware reset.</p> <p>When WFL=1 Burst access is terminated on page boundary and resume on the following page according to BL bit field configuration, because WAIT signal is not monitored from the external device</p> <p>0 the External device WAIT signal is being monitored, and it reflect the external data bus state</p> <p>1 the state of the External devices is determined internally (Fix latency mode only)</p>
3 MUM	<p>Multiplexed Mode. This bit determines the address/data multiplexed mode for asynchronous and synchronous accesses for 8 bit, 16 bit or 32 bit devices (DSZ config. dependent).</p>

*Table continues on the next page...*

**EIM\_CSnGCR1 field descriptions (continued)**

Field	Description
	<p><b>NOTE:</b> The reset value for EIM_CS0GCR1[MUM] = EIM_BOOT[2]. For EIM_CS1GCR1 - EIM_CS5GCR1 the reset value is 0.</p> <p>0 Multiplexed Mode disable 1 Multiplexed Mode enable</p>
2 SRD	<p>Synchronous Read Data. This bit field determine the read accesses mode to the External device of the chip select. The External device should be configured to the same mode as this bit implicates. SRD is cleared by a hardware reset.</p> <p><b>NOTE:</b> Sync. accesses supported only for 16/32 bit port.</p> <p>0 read accesses are in Asynchronous mode 1 read accesses are in Synchronous mode</p>
1 SWR	<p>Synchronous Write Data. This bit field determine the write accesses mode to the External device of the chip select. The External device should be configured to the same mode as this bit implicates. SWR is cleared by a hardware reset.</p> <p><b>NOTE:</b> Sync. accesses supported only for 16/32 bit port.</p> <p>0 write accesses are in Asynchronous mode 1 write accesses are in Synchronous mode</p>
0 CSEN	<p>CS Enable. This bit controls the operation of the chip select pin. CSEN is set by a hardware reset for CSGCR0 to allow external boot operation. CSEN is cleared by a hardware reset to CSGCR1-CSGCR5.</p> <p><b>NOTE:</b> Reset value for EIM_CS0GCR1 for CSEN is 1. For EIM_CS1GCR1-CS1GCR5 reset value is 0.</p> <p>0 Chip select function is disabled; attempts to access an address mapped by this chip select results in an error respond and no assertion of the chip select output 1 Chip select is enabled, and is asserted when presented with a valid access.</p>

## 20.9.2 Chip Select n General Configuration Register 2 (EIM\_CSnGCR2)

Address: 21B\_8000h base + 4h offset + (24d × i), where i=0d to 5d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		MUX16_BYP_ GRANT	0		DAP	DAE	DAPS	DAPS	DAPS	DAPS	DAPS	0		ADH	
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

**EIM\_CSnGCR2 field descriptions**

Field	Description
31–13 Reserved	This read-only field is reserved and always has the value 0.
12 MUX16_BYP_ GRANT	Muxed 16 bypass grant. This bit when asserted causes EIM to bypass the grant/ack. arbitration with NFC (only for 16 bit muxed mode accesses).  0 EIM waits for grant before driving a 16 bit muxed mode access to the memory. 1 EIM ignores the grant signal and immediately drives a 16 bit muxed mode access to the memory.
11–10 Reserved	This read-only field is reserved and always has the value 0.
9 DAP	Data Acknowledge Polarity. This bit indicates DTACK memory pin assertion state, active-low or active-high, while executing an async access using DTACK signal from the external device. DAP is cleared by a hardware reset.  0 DTACK signal is active high 1 DTACK signal is active low
8 DAE	Data Acknowledge Enable. This bit indicates external device is using DTACK pin as strobe/terminator of an async. access. DTACK signal may be used only in asynchronous single read (APR=0) or write accesses. DTACK poling start point is set by DAPS bit field. polarity of DTACK is set by DAP bit field. DAE is cleared by a hardware reset.  0 DTACK signal use is disable 1 DTACK signal use is enable
7–4 DAPS	Data Acknowledge Poling Start. This bit field determine the starting point of DTACK input signal polling. DAPS is used only in asynchronous single read or write accesses.

*Table continues on the next page...*

**EIM\_CS*n*GCR2 field descriptions (continued)**

Field	Description
	<p><b>NOTE:</b> Since DTACK is an async. signal the start point of DTACK signal polling is at least 3 cycles after the start of access.</p> <p>DAPS is cleared by a hardware reset.</p> <p>Example settings:</p> <p>0000 3 EIM clk cycle between start of access and first <math>\overline{DTACK}</math> check</p> <p>0001 4 EIM clk cycles between start of access and first <math>\overline{DTACK}</math> check</p> <p>0010 5 EIM clk cycles between start of access and first <math>\overline{DTACK}</math> check</p> <p>0111 10 EIM clk cycles between start of access and first <math>\overline{DTACK}</math> check</p> <p>1011 14 EIM clk cycles between start of access and first DTACK check</p> <p>1111 18 EIM clk cycles between start of access and first DTACK check</p>
3–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 ADH	<p>Address hold time - This bit field determine the address hold time after ADV negation when mum = 1 (muxed mode).</p> <p>When mum = 0 this bit has no effect. For read accesses the field determines when the pads direction will be switched.</p> <p><b>NOTE:</b> Reset value for EIM_CS0GCR2 for ADH is 10. For EIM_CS1GCR2-EIM_CS5GCR2 reset value is 00.</p> <p>00 0 cycle after ADV negation</p> <p>01 1 cycle after ADV negation</p> <p>10 2 cycle after ADV negation</p> <p>11 Reserved</p>

**20.9.3 Chip Select *n* Read Configuration Register 1 (EIM\_CS*n*RCR1)**

Address: 21B\_8000h base + 8h offset + (24d × i), where i=0d to 5d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				0				0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EIM\_CS*n*RCR1 field descriptions**

Field	Description
31–30 Reserved	This read-only field is reserved and always has the value 0.

*Table continues on the next page...*

**EIM\_CS $n$ RCR1 field descriptions (continued)**

Field	Description
29–24 RWSC	<p>Read Wait State Control. This bit field programs the number of wait-states, according to the settings shown below, for synchronous or asynchronous read access to the external device connected to the chip select.</p> <p>When SRD=1 and RFL=0, RWSC indicates the number of burst clock (BCLK) cycles from the start of an access, before the controller can start sample data. Since WAIT signal can be asserted one cycle before the first data can be sampled, the controller starts evaluating the WAIT signal state one cycle before, this is referred as handshake mode or variable latency mode.</p> <p>When SRD=1 and RFL=1, RWSC indicates the number of burst clock (BCLK) cycles from the start of an access, until the external device is ready for data transfer, this is referred as fix latency mode.</p> <p>When SRD=0, RFL bit is ignored, RWSC indicates the asynchronous access length and the number of EIM clock cycles from the start of access until the external device is ready for data transfer.</p> <p>RWSC is cleared by a hardware reset.</p> <p><b>NOTE:</b> The reset value for EIM_CS0RCR1, RWSC[5:0] = 0b011100. For CG1RCR1 - CS1RCR5 the reset value is 0b000000.</p> <p>Example settings:</p> <p>000000 Reserved  000001 RWSC value is 1  000010 RWSC value is 2  111101 RWSC value is 61  111110 RWSC value is 62  111111 RWSC value is 63</p>
23 Reserved	This read-only field is reserved and always has the value 0.
22–20 RADVA	<p>ADV Assertion. This bit field determines when ADV signal is asserted for synchronous or asynchronous read modes according to the settings shown below. RADVA is cleared by a hardware reset.</p> <p>Example settings:</p> <p>000 0 EIM clock cycles between beginning of access and ADV assertion  001 1 EIM clock cycles between beginning of access and ADV assertion  010 2 EIM clock cycles between beginning of access and ADV assertion  111 7 EIM clock cycles between beginning of access and ADV assertion</p>
19 RAL	Read ADV Low. This bit field determine ADV signal negation time. When RAL=1, RADVN bit field is ignored and ADV signal will stay asserted until end of access. When RAL=0 negation of ADV signal is according to RADVN bit field configuration.
18–16 RADVN	<p>ADV Negation. This bit field determines when ADV signal to memory is negated during read accesses.</p> <p>When SRD=1 (synchronous read mode), ADV negation occurs according to the following formula: (RADVN + RADVA + BCD + BCS + 1) EIM clock cycles from start of access.</p> <p>When asynchronous read mode is applied (SRD=0) and RAL=0 ADV negation occurs according to the following formula: (RADVN + RADVA + 1) EIM clock cycles from start of access. RADVN is cleared by a hardware reset.</p> <p><b>NOTE:</b> the reset value for EIM_CS0RCR1[RADVN] = 2. For EIM_CS1RCR1 - EIM_CS5RCR1, the reset value is 0b000.</p> <p><b>NOTE:</b> This field should be configured so ADV negation will occur before the end of access. For ADV negation at the same time with the end of access user should RAL bit.</p>

*Table continues on the next page...*



**EIM\_CS $n$ RCR1 field descriptions (continued)**

Field	Description
15 Reserved	This read-only field is reserved and always has the value 0.
14–12 OEA	<p>OE Assertion. This bit field determines when OE signal are asserted during read cycles (synchronous or asynchronous mode), according to the settings shown below. OEA is cleared by a hardware reset.</p> <p>In muxed mode OE assertion occurs (OEA + RADVN + RADVA + ADH +1) EIM clock cycles from start of access.</p> <p><b>NOTE:</b> The reset value for EIM_CS0RCR1[OEA] is 0b000 if EIM_BOOT[2] = 0. If EIM_BOOT[2] is 1, the reset value for EIM_CS0RCR1 is 0b010. The reset value of this field for EIM_CS1RCR1 - EIM_CS5RCR1 is 0b000.</p> <p>Example settings:</p> <p>000 0 EIM clock cycles between beginning of access and OE assertion  001 1 EIM clock cycles between beginning of access and OE assertion  010 2 EIM clock cycles between beginning of access and OE assertion  111 7 EIM clock cycles between beginning of access and OE assertion</p>
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 OEN	<p>OE Negation. This bit field determines when OE signal is negated during read cycles in asynchronous single mode only (SRD=0 &amp; APR = 0), according to the settings shown below. This bit field is ignored when SRD=1. OEN is cleared by a hardware reset.</p> <p>Example settings:</p> <p>000 0 EIM clock cycles between end of access and OE negation  001 1 EIM clock cycles between end of access and OE negation  010 2 EIM clock cycles between end of access and OE negation  111 7 EIM clock cycles between end of access and OE negation</p>
7 Reserved	This read-only field is reserved and always has the value 0.
6–4 RCSA	<p>Read CS Assertion. This bit field determines when CS signal is asserted during read cycles (synchronous or asynchronous mode), according to the settings shown below. RCSA is cleared by a hardware reset.</p> <p>Example settings:</p> <p>000 0 EIM clock cycles between beginning of read access and CS assertion  001 1 EIM clock cycles between beginning of read access and CS assertion  010 2 EIM clock cycles between beginning of read access and CS assertion  111 7 EIM clock cycles between beginning of read access and CS assertion</p>
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 RCSN	<p>Read CS Negation. This bit field determines when CS signal is negated during read cycles in asynchronous single mode only (SRD=0 &amp; APR = 0), according to the settings shown below. This bit field is ignored when SRD=1. RCSN is cleared by a hardware reset.</p> <p>Example settings:</p> <p>000 0 EIM clock cycles between end of read access and CS negation  001 1 EIM clock cycles between end of read access and CS negation  010 2 EIM clock cycles between end of read access and CS negation  111 7 EIM clock cycles between end of read access and CS negation</p>

## 20.9.4 Chip Select n Read Configuration Register 2 (EIM\_CSnRCR2)

Address: 21B\_8000h base + Ch offset + (24d × i), where i=0d to 5d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	APR	PAT			0		RL		0	RBEA			RBE	RBEN		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### EIM\_CSnRCR2 field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15 APR	Asynchronous Page Read. This bit field determine the asynchronous read mode to the external device. When APR=0, the async. read access is done as single word (where word is defined by the DSZ field). when APR=1, the async. read access executed as page read. page size is according to BL field config., RCSN,RBEN,OEN and RADVN are being ignored.  APR is cleared by a hardware reset for EIM_CS1GCR1 - EIM_CS5GCR1.  <b>NOTE:</b> SRD=0 and MUM=0 must apply when APR=1
14–12 PAT	Page Access Time. This bit field is used in Asynchronous Page Read mode only (APR=1). the initial access is set by RWSC as in regular asynchronous mode. the consecutive address assertions width determine by PAT field according to the settings shown below. when APR=0 this field is ignored.  PAT is cleared by a hardware reset for EIM_CS1GCR1 - EIM_CS5GCR1.  000 Address width is 2 EIM clock cycles 001 Address width is 3 EIM clock cycles 010 Address width is 4 EIM clock cycles 011 Address width is 5 EIM clock cycles 100 Address width is 6 EIM clock cycles 101 Address width is 7 EIM clock cycles 110 Address width is 8 EIM clock cycles 111 Address width is 9 EIM clock cycles
11–10 Reserved	This read-only field is reserved and always has the value 0.
9–8 RL	Read Latency. This bit field indicates cycle latency when executing a synchronous read operation. The fields holds the feedback clock loop delay in aclk cycle units.  This field is cleared by a hardware reset.  00 Feedback clock loop delay is up to 1 cycle for BCD = 0 or 1.5 cycles for BCD != 0 01 Feedback clock loop delay is up to 2 cycles for BCD = 0 or 2.5 cycles for BCD != 0

Table continues on the next page...

**EIM\_CS<sub>n</sub>RCR2 field descriptions (continued)**

Field	Description
10	Feedback clock loop delay is up to 3 cycles for BCD = 0 or 3.5 cycles for BCD != 0
11	Feedback clock loop delay is up to 4 cycles for BCD = 0 or 4.5 cycles for BCD != 0
7 Reserved	This read-only field is reserved and always has the value 0.
6–4 RBEA	Read BE Assertion. This bit field determines when BE signal is asserted during read cycles (synchronous or asynchronous mode), according to the settings shown below. RBEA is cleared by a hardware reset.  Example settings:  000 0 EIM clock cycles between beginning of read access and BE assertion 001 1 EIM clock cycles between beginning of read access and BE assertion 010 2 EIM clock cycles between beginning of read access and BE assertion 111 7 EIM clock cycles between beginning of read access and BE assertion
3 RBE	Read BE enable. This bit field determines if BE will be asserted during read access.  0 - BE are disabled during read access. 1- BE are enable during read access according to value of RBEA & RBEN bit fields.
2–0 RBEN	Read BE Negation. This bit field determines when BE signal is negated during read cycles in asynchronous single mode only (SRD=0 & APR=0), according to the settings shown below. This bit field is ignored when SRD=1. RBEN is cleared by a hardware reset.  Example settings:  000 0 EIM clock cycles between end of read access and BE negation 001 1 EIM clock cycles between end of read access and BE negation 010 2 EIM clock cycles between end of read access and BE negation 111 7 EIM clock cycles between end of read access and BE negation

## 20.9.5 Chip Select n Write Configuration Register 1 (EIM\_CS<sub>n</sub>WCR1)

Address: 21B\_8000h base + 10h offset + (24d × i), where i=0d to 5d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EIM\_CS<sub>n</sub>WCR1 field descriptions**

Field	Description
31 WAL	Write ADV Low. This bit field determine ADV signal negation time in write accesses. When WAL=1, WADV <sub>N</sub> bit field is ignored and ADV signal will stay asserted until end of access. When WAL=0 negation of ADV signal is according to WADV <sub>N</sub> bit field configuration.
30 WBED	Write Byte Enable Disable. When asserted this bit prevent from IPP_DO_BE_B[x] to be asserted during write accesses. This bit is cleared by hardware reset.
29–24 WWSC	<p>Write Wait State Control. This bit field programs the number of wait-states, according to the settings shown below, for synchronous or asynchronous write access to the external device connected to the chip select.</p> <p>When SWR=1 and WFL=0, WWSC indicates the number of burst clock (BCLK) cycles from the start of an access, before the memory can sample the first data. Since WAIT signal can be asserted one cycle before the first data can be sampled, the controller starts evaluating the WAIT signal state one cycle before, this is referred as handshake mode or variable latency mode.</p> <p>When SWR=1 and WFL=1, WWSC indicates the number of burst clock (BCLK) cycles from the start of an access, until the external device is ready for data transfer, this is referred as fix latency mode.</p> <p>When SWR=0, WFL bit is ignored, WWSC indicates the asynchronous access length and the number of EIM clock cycles from the start of access until the external device is ready for data transfer.</p> <p>WWSC is cleared by a hardware reset.</p> <p><b>NOTE:</b> The reset value for EIM_CS0WCR1, WWSC[5:0] = 0b011100. For EIM_CS1WCR1 - EIM_CS5WCR1, the reset value of this field is 0b000000.</p> <p>Example settings:</p> <p>000000 Reserved  000001 WWSC value is 1  000010 WWSC value is 2  000011 WWSC value is 3  111111 WWSC value is 63</p>
23–21 WADVA	<p>ADV Assertion. This bit field determines when ADV signal is asserted for synchronous or asynchronous write modes according to the settings shown below. WADVA is cleared by a hardware reset.</p> <p>Example settings:</p> <p>000 0 EIM clock cycles between beginning of access and ADV assertion  001 1 EIM clock cycles between beginning of access and ADV assertion  010 2 EIM clock cycles between beginning of access and ADV assertion  111 7 EIM clock cycles between beginning of access and ADV assertion</p>
20–18 WADV <sub>N</sub>	<p>ADV Negation. This bit field determines when ADV signal to memory is negated during write accesses.</p> <p>When SWR=1 (synchronous write mode), ADV negation occurs according to the following formula: (WADV<sub>N</sub> + WADVA + BCD + BCS + 1) EIM clock cycles.</p> <p>When asynchronous read mode is applied (SWR=0) ADV negation occurs according to the following formula: (WADV<sub>N</sub> + WADVA + 1) EIM clock cycles.</p> <p><b>NOTE:</b> Reset value for EIM_CS0WCR for WADV<sub>N</sub> is 2. For EIM_CS1WCR - EIM_CS5WCR reset value is 000.</p> <p><b>NOTE:</b> This field should be configured so ADV negation will occur before the end of access. For ADV negation at the same time as the end of access, S/W should set the WAL bit.</p>
17–15 WBEA	BE Assertion. This bit field determines when BE signal is asserted during write cycles in async. mode only (SWR=0), according to the settings shown below. BEA is cleared by a hardware reset.

*Table continues on the next page...*

**EIM\_CS<sub>n</sub>WCR1 field descriptions (continued)**

Field	Description
	<p><b>NOTE:</b> Reset value for EIM_CS0WCR for WBEA is 2. For EIM_CS1WCR - EIM_CS5WCR reset value is 000.</p> <p>Example settings:</p> <p>000 0 EIM clock cycles between beginning of access and BE assertion</p> <p>001 1 EIM clock cycles between beginning of access and BE assertion</p> <p>010 2 EIM clock cycles between beginning of access and BE assertion</p> <p>111 7 EIM clock cycles between beginning of access and BE assertion</p>
14–12 WBEN	<p>BE[3:0] Negation. This bit field determines when BE[3:0] bus signal is negated during write cycles in async. mode only (SWR=0), according to the settings shown below. This bit field is ignored when SWR=1. BEN is cleared by a hardware reset.</p> <p><b>NOTE:</b> Reset value for EIM_CS0WCR for WBEN is 2. For EIM_CS1WCR - EIM_CS5WCR reset value is 000.</p> <p>Example settings:</p> <p>000 0 EIM clock cycles between end of access and WE negation</p> <p>001 1 EIM clock cycles between end of access and WE negation</p> <p>010 2 EIM clock cycles between end of access and WE negation</p> <p>111 7 EIM clock cycles between end of access and WE negation</p>
11–9 WEA	<p>WE Assertion. This bit field determines when WE signal is asserted during write cycles (synchronous or asynchronous mode), according to the settings shown below. This bit field is ignored when executing a read access to the external device. WEA is cleared by a hardware reset.</p> <p><b>NOTE:</b> Reset value for EIM_CS0WCR for WEA is 2. For EIM_CS1WCR - EIM_CS5WCR reset value is 000.</p> <p>Example settings:</p> <p>000 0 EIM clock cycles between beginning of access and WE assertion</p> <p>001 1 EIM clock cycles between beginning of access and WE assertion</p> <p>010 2 EIM clock cycles between beginning of access and WE assertion</p> <p>111 7 EIM clock cycles between beginning of access and WE assertion</p>
8–6 WEN	<p>WE Negation. This bit field determines when WE signal is negated during write cycles in asynchronous mode only (SWR=0), according to the settings shown below. This bit field is ignored when SWR=1. WEN is cleared by a hardware reset.</p> <p><b>NOTE:</b> Reset value for EIM_CS0WCR for WEN is 2. For EIM_CS1WCR - EIM_CS5WCR reset value is 000.</p> <p>Example settings:</p> <p>000 0 EIM clock cycles between beginning of access and WE assertion</p> <p>001 1 EIM clock cycles between beginning of access and WE assertion</p> <p>010 2 EIM clock cycles between beginning of access and WE assertion</p> <p>111 7 EIM clock cycles between beginning of access and WE assertion</p>
5–3 WCSA	<p>Write CS Assertion. This bit field determines when CS signal is asserted during write cycles (synchronous or asynchronous mode), according to the settings shown below. This bit field is ignored when executing a read access to the external device. WCSA is cleared by a hardware reset.</p> <p>Example settings:</p> <p>000 0 EIM clock cycles between beginning of write access and CS assertion</p>

*Table continues on the next page...*

**EIM\_CS<sub>n</sub>WCR1 field descriptions (continued)**

Field	Description
	001 1 EIM clock cycles between beginning of write access and CS assertion 010 2 EIM clock cycles between beginning of write access and CS assertion 111 7 EIM clock cycles between beginning of write access and CS assertion
2–0 WCSN	Write CS Negation. This bit field determines when CS signal is negated during write cycles in asynchronous mode only (SWR=0), according to the settings shown below. This bit field is ignored when SWR=1. WCSN is cleared by a hardware reset.  Example settings:  000 0 EIM clock cycles between end of read access and CS negation 001 1 EIM clock cycles between end of read access and CS negation 010 2 EIM clock cycles between end of read access and CS negation 111 7 EIM clock cycles between end of read access and CS negation

**20.9.6 Chip Select n Write Configuration Register 2 (EIM\_CS<sub>n</sub>WCR2)**

Address: 21B\_8000h base + 14h offset + (24d × i), where i=0d to 5d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															WBCDD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EIM\_CS<sub>n</sub>WCR2 field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 WBCDD	Write Burst Clock Divisor Decrement. If this bit is asserted and BCD value is 0 sync. write access will be preformed as if BCD value is 1. When this bit is negated or BCD value is not 0 this bit has no affect.  This bit is cleared by hardware reset.

## 20.9.7 EIM Configuration Register (EIM\_WCR)

Address: 21B\_8000h base + 90h offset = 21B\_8090h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Hardware Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FRUN_ACLK_EN	WD OG_ LIMIT			WD OG_ EN	0		INTPOL	INTEN	CONT_BCLK_SEL	GBCD	BCM
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
Hardware Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

### EIM\_WCR field descriptions

Field	Description
31–12 Reserved	Reserved This read-only field is reserved and always has the value 0.
11 FRUN_ACLK_EN	Free run ACLK enable
10–9 WD OG_ LIMIT	Memory Watch Dog (WDog) cycle limit. This bit field determines the number of BCLK cycles (ACLK cycles in dtrack mode) before the wdog counter terminates the access and send an error response to the master.  00 128 BCLK cycles 01 256 BCLK cycles 10 512 BCLK cycles 11 1024 BCLK cycles
8 WD OG_ EN	Memory WDog enable. This bit controls the operation of the wdog counter that terminates the EIM access.  0 Memory WDog is Disabled 1 Memory WDog is Enabled

Table continues on the next page...

**EIM\_WCR field descriptions (continued)**

Field	Description
7–6 Reserved	This read-only field is reserved and always has the value 0.
5 INTPOL	Interrupt Polarity. This bit field determines the polarity of the external device interrupt.  0 External interrupt polarity is active low 1 External interrupt polarity is active high
4 INTEN	Interrupt Enable. When this bit is set the External signal RDY_INT as active interrupt. When interrupt occurs, INT bit at the WCR will be set and t EIM_EXT_INT signal will be asserted correspondingly. This bit is cleared by a hardware reset.  0 External interrupt Disable 1 External interrupt Enable
3 CONT_BCLK_SEL	Continuous BCLK select  When this bit is set BCLK pin output continuous clock. Otherwise, BCLK will output clock only when nesserary.  0 BCLK When nesserary 1 BCLK Continuous
2–1 GBCD	General Burst Clock Divisor. When BCM bit is set, this bit field contains the value used to program the burst clock divisor for Continuous BCLK generation. The other BCD bit fields for each chip select are ignored. It is used to divide the internal AXI bus frequency. When BCM=0 GB CD bit field has no influence. GB CD is cleared by a hardware reset.  00 Divide EIM clock by 1 01 Divide EIM clock by 2 10 Divide EIM clock by 3 11 Divide EIM clock by 4
0 BCM	Burst Clock Mode. This bit selects the burst clock mode of operation. It is used for system debug mode. BCM is cleared by a hardware reset.  <b>NOTE:</b> The BCLK frequency in this mode is according to GB CD bit field.  <b>NOTE:</b> The BCLK phase is opposite to the EIM clock in this mode if GB CD is 0.  <b>NOTE:</b> This bit should be used only in async. accesses. No sync access can be executed if this bit is set.  <b>NOTE:</b> When this bit is set bcd field shouldn't be configured to 0.  0 The burst clock runs only when accessing a chip select range with the SWR/SRD bits set. When the burst clock is not running it remains in a logic 0 state. When the burst clock is running it is configured by the BCD and BCS bit fields in the chip select Configuration Register. 1 The burst clock runs whenever ACLK is active (independent of chip select configuration)



## 20.9.8 DLL Control Register (EIM\_DCR)

Address: 21B\_8000h base + 94h offset = 21B\_8094h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	DLL_CTRL_REF_UPDATE_INT				DLL_CTRL_SLV_UPDATE_INT					DLL_CTRL_REF_INITIAL_VAL							
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DLL_CTRL_SLV_OVERRIDE_VAL								DLL_CTRL_SLV_OVERRIDE	DLL_CTRL_GATE_UPDATE	DLL_CTRL_SLV_OFFSET		DLL_CTRL_SLV_OFFSET_DEC	DLL_CTRL_SLV_FORCE_UPD	DLL_CTRL_RESET	DLL_CTRL_ENABLE
W											DLL_CTRL_SLV_OFFSET					
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### EIM\_DCR field descriptions

Field	Description
31–28 DLL_CTRL_REF_UPDATE_INT	Reference DLL Update Interval  DLL control loop update interval. The interval cycle is $(2 + \text{REF\_UPDATE\_INT}) \times \text{ref\_clock}$ . By default, the DLL control loop shall update every two ref_clock cycles. It should be noted that increasing the reference delay-line update interval reduces the ability of the DLL to adjust to fast changes in conditions that may effect the delay (such as voltage and temperature)
27–23 DLL_CTRL_SLV_UPDATE_INT	Slave DLL Update Interval  If default 0 is used, it means 256 cycles of ref_clock. A value of 0x0f results in 15 cycles and so on. Note that software can always cause an update of the slave-delay line using the SLV_FORCE_UPDATE register. Note that the slave delay line will also update automatically when the reference DLL transitions to a locked state (from an un-locked state).
22–16 DLL_CTRL_REF_INITIAL_VAL	This field is used to select the initial value of reference chain before DLL enabled. It's recommended to set the initial value close to the locked value to accelerate the locking.
15–9 DLL_CTRL_SLV_OVERRIDE_VAL	When SLV_OVERRIDE=1 This field is used to select 1 of 128 physical taps manually. A value of 0 selects tap 1, and a value of 0x7f selects tap 128.
8 DLL_CTRL_SLV_OVERRIDE	Set this bit to 1 to Enable manual override for slave delay chain using SLV_OVERRIDE_VAL; to set 0 to disable manual override. This feature does not require the DLL to be enabled using the ENABLE bit. In fact to reduce power, if SLV_OVERRIDE is used, it is recommended to disable the DLL with ENABLE=0

Table continues on the next page...

**EIM\_DCR field descriptions (continued)**

Field	Description
7 DLL_CTRL_ GATE_UPDATE	Set this bit to 1 to force DLL not update from now on. Since when clock exists, glitches might appear during update. This bit is used by software if we met such kind of condition. Set it to 0 to let DLL update automatically
6-4 DLL_CTRL_ SLV_OFFSET	OFFSET value for DLL_CTRL_SLV_SEL
3 DLL_CTRL_ SLV_OFFSET_ DEC	Slave Chain Offset Decrease  Decrease(or increase) the value defined by DLL_CTRL_SLV_OFFSET when calculating DLL_STS_SLV_SEL  0 DLL_STS_SLV_SEL = DLL_STS_REF_SEL + DLL_CTRL_SLV_OFFSET 1 DLL_STS_SLV_SEL = DLL_STS_REF_SEL - DLL_CTRL_SLV_OFFSET
2 DLL_CTRL_ SLV_FORCE_ UPD	Setting this bit to 1, forces the slave delay line to update to the DLL calibrated value immediately. The slave delay line shall update automatically based on the SLV_UPDATE_INT interval or when a DLL lock condition is sensed. Subsequent forcing of the slave-line update can only occur if SLV_FORCE_UP is set back to 0 and then asserted again (edge triggered). Be sure to use it when EIM is idle.
1 DLL_CTRL_ RESET	DLL Reset Bit  Setting this bit to 1 force a reset on DLL. This will cause the DLL to lose lock and re-lock. This signal is used by the DLL as edge-sensitive, so in order to create a subsequent reset, RESET must be taken low and then asserted again
0 DLL_CTRL_ ENABLE	Set this bit to 1 to enable the DLL and delay chain; otherwise; set to 0 to bypasses DLL. Note that using the slave delay line override feature with SLV_OVERRIDE and SLV_OVERRIDE_VAL, the DLL does not need to be enabled

**20.9.9 DLL Status Register (EIM\_DSR)**

Address: 21B\_8000h base + 98h offset = 21B\_8098h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DLL_STS_REF_SEL								DLL_STS_SLV_SEL							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EIM\_DSR field descriptions**

Field	Description
31–16 Reserved	Reserved  This read-only field is reserved and always has the value 0.
15–9 DLL_STS_REF_SEL	Reference delay line select taps. Be noted this is encoded by 7 bits for 127taps.
8–2 DLL_STS_SLV_SEL	Slave delay line select status. This is the instant value generated from reference chain. Since only when ref_lock is detected can the reference chain get updated, this value should be the right value next be update to the slave line when reference is locked.
1 DLL_STS_REF_LOCK	Reference DLL lock status. This signifies that the DLL has detected and locked to feedback BCLK, allowing the slave delay-line to perform programmed clock delays
0 DLL_STS_SLV_LOCK	Slave delay-line lock status. This signifies that a valid delay has been set to the slave-delay line and that the slave-delay line is implementing the programmed delay value

**20.9.10 EIM IP Access Register (EIM\_WIAR)**

Address: 21B\_8000h base + 9Ch offset = 21B\_809Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W													ACLK_EN	ERRST	INT	IPS_ACK
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

**EIM\_WIAR field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 ACLK_EN	ACLK enable. This bit gates the ACLK for the EIM except from FFs that get ipg_aclk_s. After reset ACLK is enabled.  0 ACLK is disabled 1 ACLK is enabled

*Table continues on the next page...*

**EIM\_WIAR field descriptions (continued)**

Field	Description
3 ERRST	<p>READY After Reset. This bit controls the initial ready/busy status for external devices on CS0 immediately after hardware reset. This is a sticky bit which is cleared once the RDY_INT signal is asserted by the external device.</p> <p>When ERRST = 1 the first fetch access from EIM to the external device located on CS0 will be pending until RDY_INT signal indicates that the external device is ready, then EIM will execute the access.</p> <p><b>NOTE:</b> Reset value for ERRST is EIM_BOOT[4].</p> <p>0 RDY_INT After Reset Disable 1 RDY_INT After Reset Enable</p>
2 INT	<p>Interrupt. This bit indicates interrupt assertion by an external device according to RDY_INT signal. When polling this bit, INT=0 indicates interrupt not occurred and INT=1 indicates assertion of the external device interrupt. This bit is cleared by a hardware reset.</p>
1 IPS_ACK	<p>IPS ACK. The EIM is ready for ips access. There is no active AXI access and no new AXI access is accepted till this bit is cleared. This bit is cleared by the master after it completes the ips accesses.</p> <p>0 Master cannot access ips. 1 Master can access ips.</p>
0 IPS_REQ	<p>IPS request. The Master requests to access one of the IPS registers. During such access the EIM should not perform any AXI/memory accesses. The EIM finishes the AXI accesses that already starts and asserts the IPS_ACK bit.</p> <p>0 No Master requests ips access 1 Master requests ips access</p>

**20.9.11 Error Address Register (EIM\_EAR)**

Address: 21B\_8000h base + A0h offset = 21B\_80A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EIM\_EAR field descriptions**

Field	Description
31–0 Error_ADDR	Error Address. This bit field holds the AXI address of the last access that caused error. This register is read only register.

# Chapter 21

## Enhanced LCD Interface (eLCDIF)

### 21.1 Overview

The eLCDIF is a general purpose display controller used to drive a wide range of display devices varying in size and capability.

Many of these displays have had an asynchronous parallel MPU interface for command and data transfer to an integrated frame buffer. There are other popular displays that support moving pictures and require the RGB interface mode (called DOTCLK interface in this document) or the VSYNC mode for high-speed data transfers. In addition to these displays, it is also common to provide support for digital video encoders that accept ITU-R BT.656 format 4:2:2 YCbCr digital component video and convert it to analog TV signals. The eLCDIF block supports these interfaces by providing fully programmable functionality.

The block has several major features:

- Bus master interface to source frame buffer data for display refresh. This interface can also be used to drive data for "Smart" displays.
- PIO interface to manage data transfers between "Smart" displays and SoC.
- 8/16/18/24/32 bit LCD data bus support available depending on I/O mux options.
- Programmable timing and parameters for MPU, VSYNC and DOTCLK LCD interfaces to support a wide variety of displays.
- ITU-R BT.656 mode (called Digital Video Interface or DVI mode here) including progressive-to-interlace feature and RGB to YCbCr 4:2:2 color space conversion to support 525/60 and 625/50 operation.

### 21.2 External Signals

The following table describes the external signals of LCD:

**Table 21-1. LCD External Signals**

Signal	Description	Pad	Mode	Direction
LCD_BUSY	Busy Signal	LCD_RESET	ALT2	I
LCD_CLK	Clock signal	LCD_CLK	ALT0	I
LCD_CS	Chip select	LCD_HSYNC	ALT2	O
LCD_DATA00	Data signal	KEY_COL0	ALT2	IO
		LCD_DAT0	ALT0	
LCD_DATA01	Data signal	KEY_ROW0	ALT2	IO
		LCD_DAT1	ALT0	
LCD_DATA02	Data signal	KEY_COL1	ALT2	IO
		LCD_DAT2	ALT0	
LCD_DATA03	Data signal	KEY_ROW1	ALT2	IO
		LCD_DAT3	ALT0	
LCD_DATA04	Data signal	KEY_COL2	ALT2	IO
		LCD_DAT4	ALT0	
LCD_DATA05	Data signal	KEY_ROW2	ALT2	IO
		LCD_DAT5	ALT0	
LCD_DATA06	Data signal	KEY_COL3	ALT2	IO
		LCD_DAT6	ALT0	
LCD_DATA07	Data signal	KEY_ROW3	ALT2	IO
		LCD_DAT7	ALT0	
LCD_DATA08	Data signal	KEY_COL4	ALT2	IO
		LCD_DAT8	ALT0	
LCD_DATA09	Data signal	KEY_ROW4	ALT2	IO
		LCD_DAT9	ALT0	
LCD_DATA10	Data signal	KEY_COL5	ALT2	IO
		LCD_DAT10	ALT0	
LCD_DATA11	Data signal	KEY_ROW5	ALT2	IO
		LCD_DAT11	ALT0	
LCD_DATA12	Data signal	KEY_COL6	ALT2	IO
		LCD_DAT12	ALT0	
LCD_DATA13	Data signal	KEY_ROW6	ALT2	IO
		LCD_DAT13	ALT0	
LCD_DATA14	Data signal	KEY_COL7	ALT2	IO
		LCD_DAT14	ALT0	
LCD_DATA15	Data signal	KEY_ROW7	ALT2	IO
		LCD_DAT15	ALT0	
LCD_DATA16	Data signal	EPDC_PWRCTRL0	ALT2	IO
		LCD_DAT16	ALT0	
LCD_DATA17	Data signal	EPDC_PWRCTRL1	ALT2	IO
		LCD_DAT17	ALT0	

*Table continues on the next page...*

**Table 21-1. LCD External Signals (continued)**

Signal	Description	Pad	Mode	Direction
LCD_DATA18	Data signal	EPDC_PWRCTRL2	ALT2	IO
		LCD_DAT18	ALT0	
LCD_DATA19	Data signal	EPDC_PWRCTRL3	ALT2	IO
		LCD_DAT19	ALT0	
LCD_DATA20	Data signal	EPDC_PWRCOM	ALT2	IO
		LCD_DAT20	ALT0	
LCD_DATA21	Data signal	EPDC_PWRINT	ALT2	IO
		LCD_DAT21	ALT0	
LCD_DATA22	Data signal	EPDC_PWRSTAT	ALT2	IO
		LCD_DAT22	ALT0	
LCD_DATA23	Data signal	EPDC_PWRWAKEUP	ALT2	IO
		LCD_DAT23	ALT0	
LCD_DATA24	Data signal	EPDC_D0	ALT2	IO
LCD_DATA25	Data signal	EPDC_D1	ALT2	IO
LCD_DATA26	Data signal	EPDC_D2	ALT2	IO
LCD_DATA27	Data signal	EPDC_D3	ALT2	IO
LCD_DATA28	Data signal	EPDC_D4	ALT2	IO
LCD_DATA29	Data signal	EPDC_D5	ALT2	IO
LCD_DATA30	Data signal	EPDC_D6	ALT2	IO
LCD_DATA31	Data signal	EPDC_D7	ALT2	IO
LCD_ENABLE	Enable signals	LCD_ENABLE	ALT0	IO
LCD_HSYNC	HSYNC signal	LCD_HSYNC	ALT0	I
LCD_RD_E	RD_E signal	LCD_ENABLE	ALT2	IO
LCD_RESET	Reset signal	LCD_RESET	ALT0	IO
LCD_RS	RS signal	LCD_VSYNC	ALT2	O
LCD_VSYNC	VSYNC signal	LCD_VSYNC	ALT0	I
LCD_WR_RWN	WR signal	LCD_CLK	ALT2	IO

## 21.3 Clocks

The following table describes the clock sources for eLCDIF. Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 21-2. eLCDIF Clocks**

Clock name	Clock Root	Description
apb_clk	lcdif_axi_clk_root	AXI clock
pix_clk	lcdif_pix_clk_root	Pixel clock

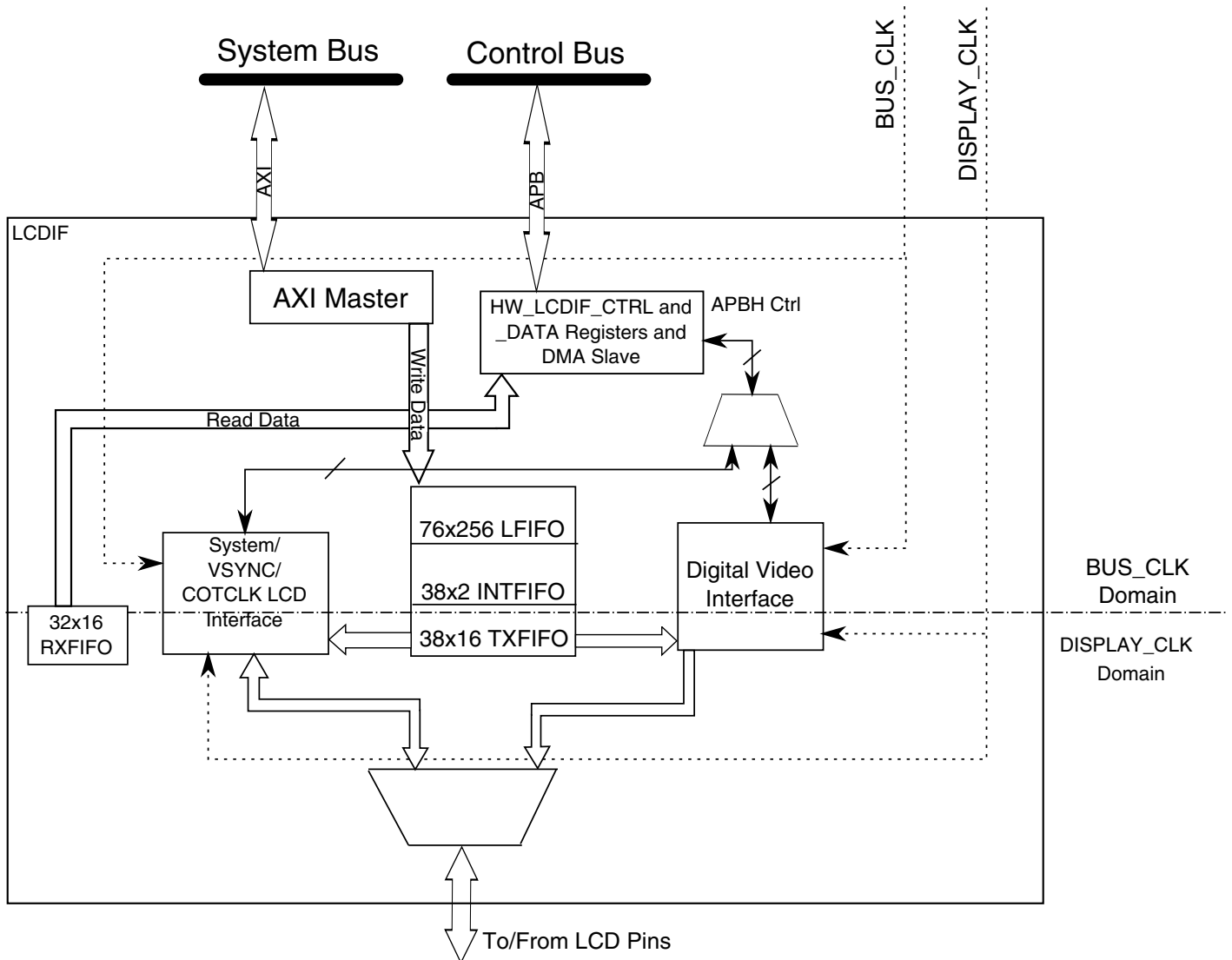
## 21.4 Functional Description

[Bus Interface Mechanisms](#) through [Initializing the eLCDIF](#), describe the internal pipeline for the MPU write/read interface, VSYNC, DOTCLK, and DVI interfaces. Differences for each mode are then described in separate sections, as follows:

- [MPU Interface](#)
- [VSYNC Interface](#)
- [DOTCLK Interface](#)
- [ITU-R BT.656 Digital Video Interface \(DVI\)](#)

eLCDIF pin usage by interface mode is described in [eLCDIF Pin Usage by Interface Mode](#).





**Figure 21-1. Top-Level Block Diagram of eLCDIF subsystem**

### 21.4.1 Bus Interface Mechanisms

The LCDIF module has memory-mapped control, data and status registers. It provides several interfaces to transfer data between the display and SoC.

The bus master interface is used to initiate the requests to transfer data from external memory to the display. It is completely autonomous, or no CPU intervention is required, to manage the cyclical nature of refreshing standard display types. Bus mastering can also be used for MPU mode data writes.

The PIO interface is used to interface to "Smart" displays to transfer frame buffer data and control information to/from the external display. The host CPU executes display drivers to manage the display solution.

The following sections describe the system bus interface mechanisms.

### **21.4.1.1 Bus Master Operation in Write/Display Modes**

The eLCDIF block has a bus master interface that initiates requests for data to drive the display. The LCDIF\_MASTER bit must be set to 1 to enable the bus master interface. Software should program all control registers required to transfer the frame sequence.

In the MPU and VSYNC modes, single frames are transferred. When a complete frame is transferred, eLCDIF enters idle and clears the RUN bit in the CTRL register. For subsequent frame transmission, the eLCDIF setup sequence should be repeated.

The DOTCLK and DVI modes are used to refresh the display at the desired refresh rate and resolution. These modes are used to drive displays that do not integrate a display buffer memory. When the display is refreshed, the eLCDIF will automatically update the LCDIF\_CUR\_BUF\_ADDR register with the value in LCDIF\_NEXT\_BUF\_ADDR at the end of current frame and start fetching the next frame from the new address. If the LCDIF\_NEXT\_BUF\_ADDR register was not updated within a frame refresh cycle, eLCDIF will keep transmitting the last frame until a new value is programmed into that register.

eLCDIF also provides the capability of interlacing a progressive frame by fetching odd lines in the first field and then fetching even lines in the second field. This feature can be used in the DVI mode and can be turned on by setting the INTERLACE\_FIELDS bit in the LCDIF\_CTRL1 register.

### **21.4.1.2 System Bus Master Performance**

The performance of the eLCDIF block can be controlled by changing the burst length and the outstanding cycle issuing capability depending on the memory bandwidth requirements. Two fields in the LCDIF\_CTRL2 register will throttle system memory requests. The LCDIF\_CTRL2\_OUTSTANDING\_REQS field will control how many requests the eLCDIF can have in flight on any given clock cycle. This should be programmed based on the expected system bus latency for returned read data. Also, the LCDIF\_CTRL2\_BURST\_LEN\_8 bit will set the number of 64 bit words requested for each eLCDIF system bus request to either 8 or 16 QWORDS. Generally, 4 outstanding requests of length 16 will provide enough performance to drive any standard display resolution. These configuration bits are intended to change the access pattern of the eLCDIF to optimize system bus throughput when other system masters will contend for system memory resources.

The LCDIF\_THRES register can also be used to optimize bus throughput and power consumption.

The LCDIF\_THRES\_PANIC value can be used to raise the priority of requests initiated by the eLCDIF to alter how the eLCDIF requests are arbitrated by the system bus infrastructure. The panic output control signal is raised when the number of 32bpp pixel equivalents in the LFIFO is less than this programmed value. Since the LFIFO is arranged as a 256x64bit quadword FIFO, it contains two 32bpp pixels per quadword, or 512 32bpp pixels total. To set the panic output when 3/4s of the LFIFO is empty, set the LCDIF\_THRES\_PANIC value to  $3/4 * 512$ , or 128. The panic signal output is used to assess higher priority to eLCDIF system requests to avoid eLCDIF under run errors during periods of high system bandwidth utilization.

The features available with the LCDIF\_THRES register require support from system clocking and dynamic priority control. Refer to the appropriate block documentation to assess the system support for these features.

## 21.4.2 Write Data Path

eLCDIF supports raster based frame buffers and there is no support for tiled buffers.

There are several options to accommodate endianness of display buffers in memory before the data is processed for the external display. The INPUT\_DATA\_SWIZZLE field in the LCDIF\_CTRL register provides the following options for data word multiplexing:

```
00 (0): No swizzle (little-endian)
01 (1): Swap bytes 0 and 3, swap bytes 1 and 2 (big-endian)
10 (2): Swap half-words
11 (3): Swap bytes within each half-word
```

The WORD\_LENGTH field of LCDIF\_CTRL register indicates the input data/pixel format. LCDIF\_TRANSFER\_COUNT register denotes how much data is contained in each frame. The H\_COUNT field of this register indicates the number of pixels per line and V\_COUNT indicates the total number of lines per frame. A special bit field in the LCDIF\_CTRL1 register, called the BYTE\_PACKING\_FORMAT, can be used to specify which bytes within the 32-bit word are going to be valid. For example, if the entire 32-bit word is valid, BYTE\_PACKING\_FORMAT should be set to 0xF, if only lower 3 bytes of each word in the frame buffer are valid, then BYTE\_PACKING\_FORMAT should be set to 0x7.

The LCD\_DATABUS\_WIDTH field in LCDIF\_CTRL register suggests the width of the bus going to the external display controller. There is an option to source all 32 bits of the input word and transfer it to the output I/O display interface. Refer to the system I/O muxing options for support of this feature. If the LCD\_DATABUS\_WIDTH is not the

same as WORD\_LENGTH, eLCDIF will perform RGB to RGB color space conversion. For example, if the input frame has fewer bits per pixel than the display, as in a 16 bpp input frame going to 24 bpp LCD, eLCDIF will pad the MSBs of each color to the LSBs of the same color for each pixel. If the input frame has more bits per pixel than the display, for example, 24 bpp input frame going to 16 bpp LCD, eLCDIF will drop the LSBs of each color channel to convert to the lower color depth. eLCDIF also has the capability to support delta pixel displays by swizzling the R, G and B colors of each pixel in the odd and even lines of the frame separately by programming the ODD\_LINE\_PATTERN and the EVEN\_LINE\_PATTERN bit fields. This operation occurs after the RGB-to-RGB color space conversion operation.

eLCDIF also supports RGB to YCbCr 4:2:2 color space conversion. This is useful in the DVI mode since the TV encoder requires input in YCbCr 4:2:2 format. The LCDIF\_CSC\* registers have complete programmability over the CSC coefficients and offsets. The values must be written into these registers in the signed two's complement format.

The following list shows how the different input/output combinations can be obtained:

- WORD\_LENGTH=1 indicates that the input is 8-bit data. This is most likely going to be used for sending commands in MPU interface, or maybe a gray scale image. Any combination of BYTE\_PACKING\_FORMAT [3:0] is permissible.

Limitation: H\_COUNT must be a multiple of the sum of BYTE\_PACKING\_FORMAT [3], BYTE\_PACKING\_FORMAT [2], BYTE\_PACKING\_FORMAT [1] and BYTE\_PACKING\_FORMAT [0].  
LCD\_DATABUS\_WIDTH must be 1, indicating an 8-bit data bus.

- WORD\_LENGTH=0 implies the input frame buffer is RGB 16 bits per pixel. DATA\_FORMAT\_16\_BIT field determines the pixels are RGB 555 or RGB 565.

Limitation: BYTE\_PACKING\_FORMAT [3:0] should be 0x3 or 0xC if there is only one pixel per word. If there are two pixels per word, it should be 0xF and H\_COUNT will be restricted to be a multiple of 2 pixels.

- WORD\_LENGTH=2 indicates that input frame buffer is RGB 18 bits per pixel, that is, RGB 666. The valid RGB values can be left-aligned or right-aligned within a 32-bit word. The alignment of the valid 18 bits within a word is indicated by the DATA\_FORMAT\_18\_BIT bit.

Limitation: BYTE\_PACKING\_FORMAT can be 0x7, 0xE or 0xF. Packed pixels are not supported in this case. H\_COUNT can be any number.

- WORD\_LENGTH=3 indicates that the input frame-buffer is RGB 24 bits per pixel (RGB 888). If BYTE\_PACKING\_FORMAT [3:0] is 0x7, it indicates that there is only one pixel per 32-bit word and there is no restriction on H\_COUNT. This is also

the option that provides 32 bit output depending on the I/O muxing options available. The fourth byte, or bits [31:24], and connected to the I/Os if this muxing is available in the chip package.

Limitation: If BYTE\_PACKING\_FORMAT [3:0] is 0xF, it indicates that the pixels are packed, that is, there are 4 pixels in 3 words or 12 bytes and H\_COUNT must be a multiple of 4 pixels.

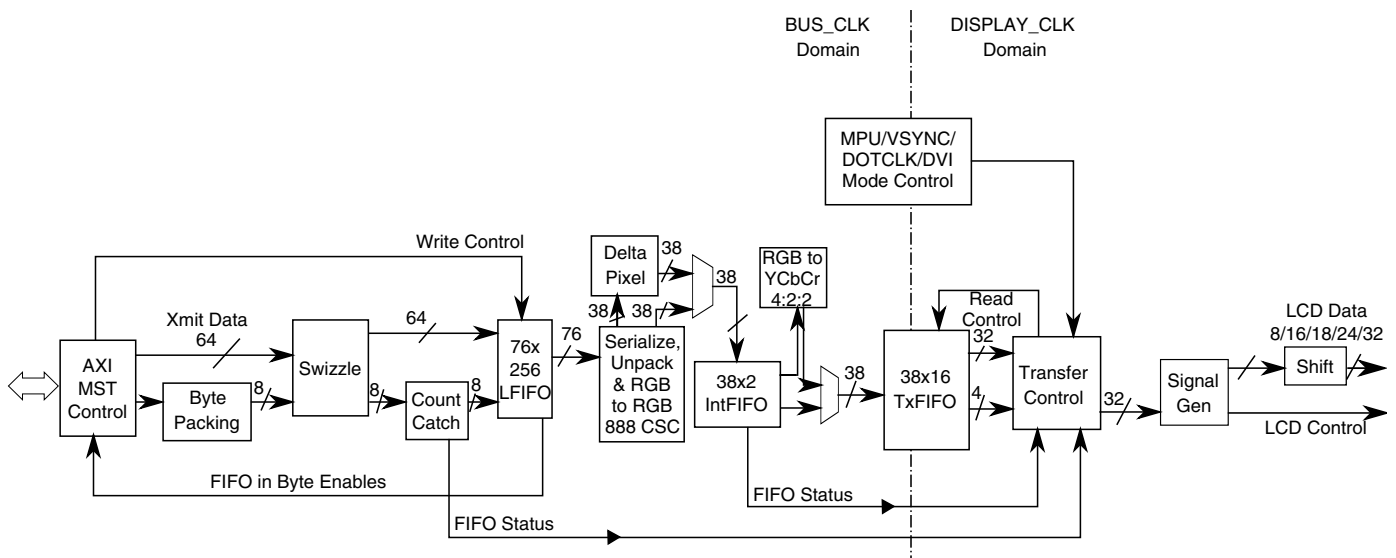
- YCBCR422\_INPUT=1 implies that the input frame is in YCbCr 4:2:2 format. BYTE\_PACKING\_FORMAT must be 0xF.

Limitation: LCD\_DATABUS\_WIDTH must be 8-bit and H\_COUNT must be a multiple of 2 pixels.

ODD\_LINE\_PATTERN and EVEN\_LINE\_PATTERN must be 0 when any of RGB\_TO\_YCBCR422\_CSC or INTERLACE\_FIELDS or YCBCR422\_INPUT bits is 1.

After the RGB to RGB or RGB to YCbCr 4:2:2 color space conversions, there is one more opportunity to swizzle the data before sending it out to the display or the encoder. This can be done with the CSC\_DATA\_SWIZZLE field in the LCDIF\_CTRL register, and it provides the same options as the INPUT\_DATA\_SWIZZLE register.

Finally, there is an option to shift the output data before sending it out to the display. This is done based on the SHIFT\_DIR and SHIFT\_NUM\_BITS fields in LCDIF\_CTRL register.

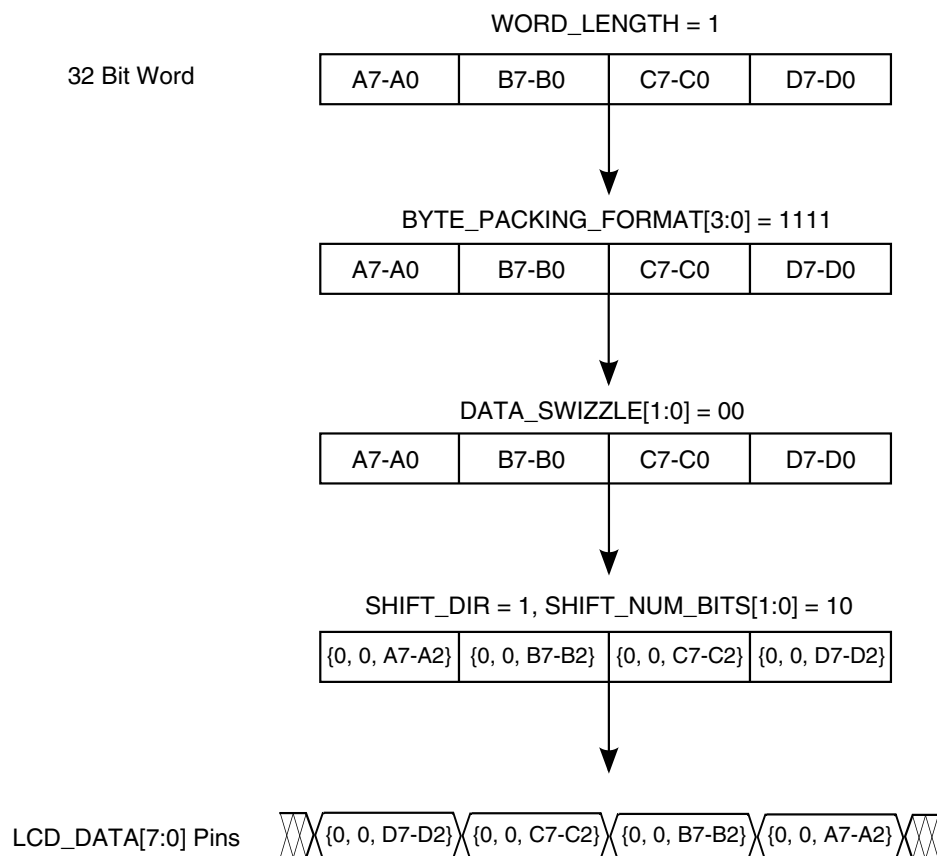


**Figure 21-2. General Operations in Write Data Path**

## Functional Description

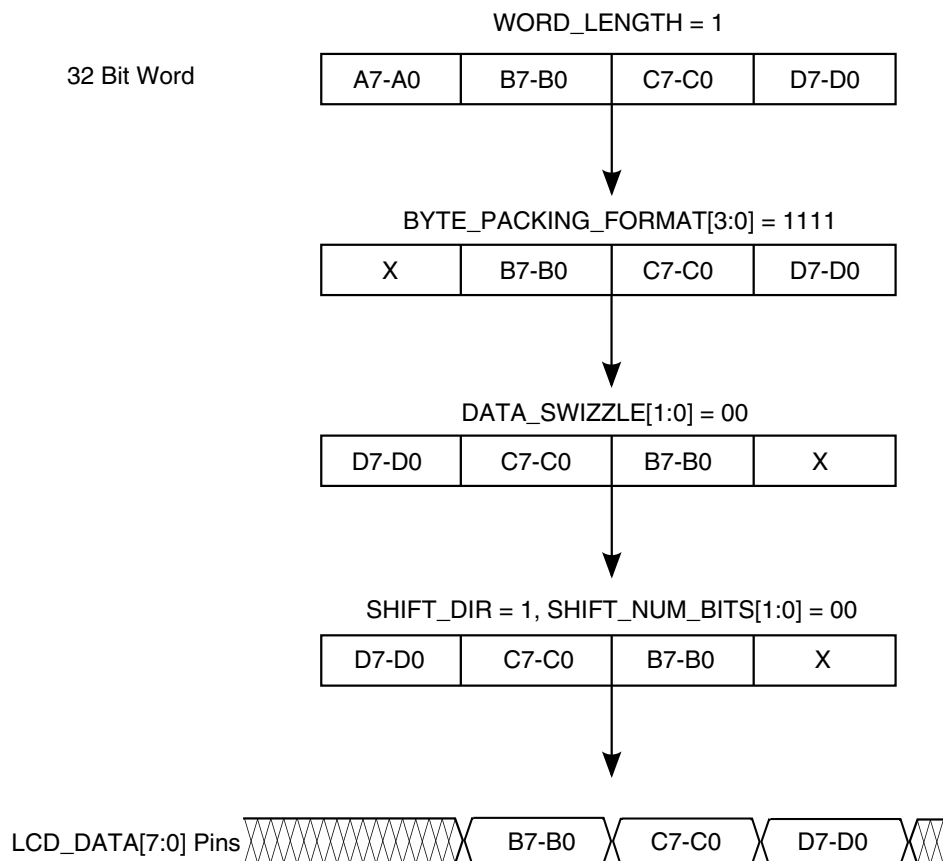
The examples in the following figures illustrate some different combinations of register programming for write mode. Assume that the data transferred over the system bus within a 32 bit word is organized as {A7-A0, B7-B0, C7-C0, D7-D0} in 8-bit mode and {A15-A0, B15-B0} in 16-bit mode.

In this example, all 32 bits of the input word are transferred out over an 8 bit display bus. Each byte within the 32 bit word is shifted to the right with zeros appended to I/O bits D[7:6]. The input data bits [7:2] are shifted to the right by 2 bits and presented on the D[5:0].



**Figure 21-3. Register programming for write mode**

In this 8 bit display interface example, one byte of the input word is deleted and not transferred over the external 8 bit display interface. This mode could be used to transfer 24bpp pixels over the 8 bit interface. In this case, the 4th unused byte is not transferred.



**Figure 21-4. Register programming for write mode**

The following example uses a 16 bit display interface. Each 16 bit half word is shifted to the right by two bits with zeros appended to the most significant two bits.

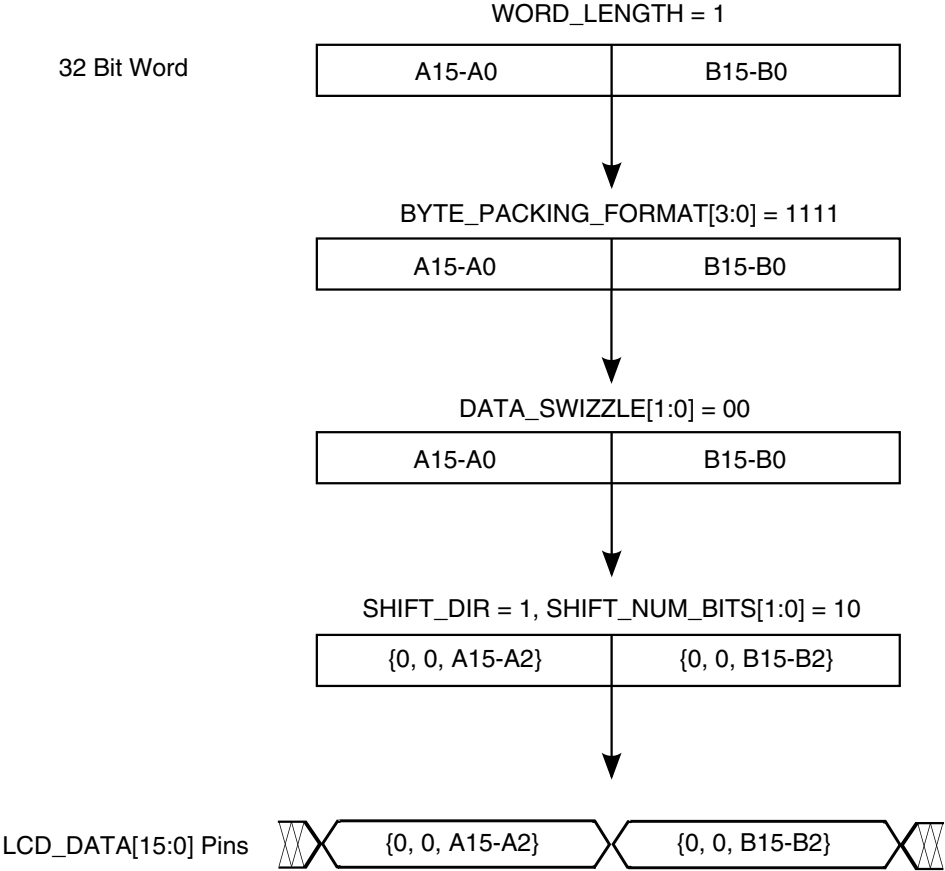
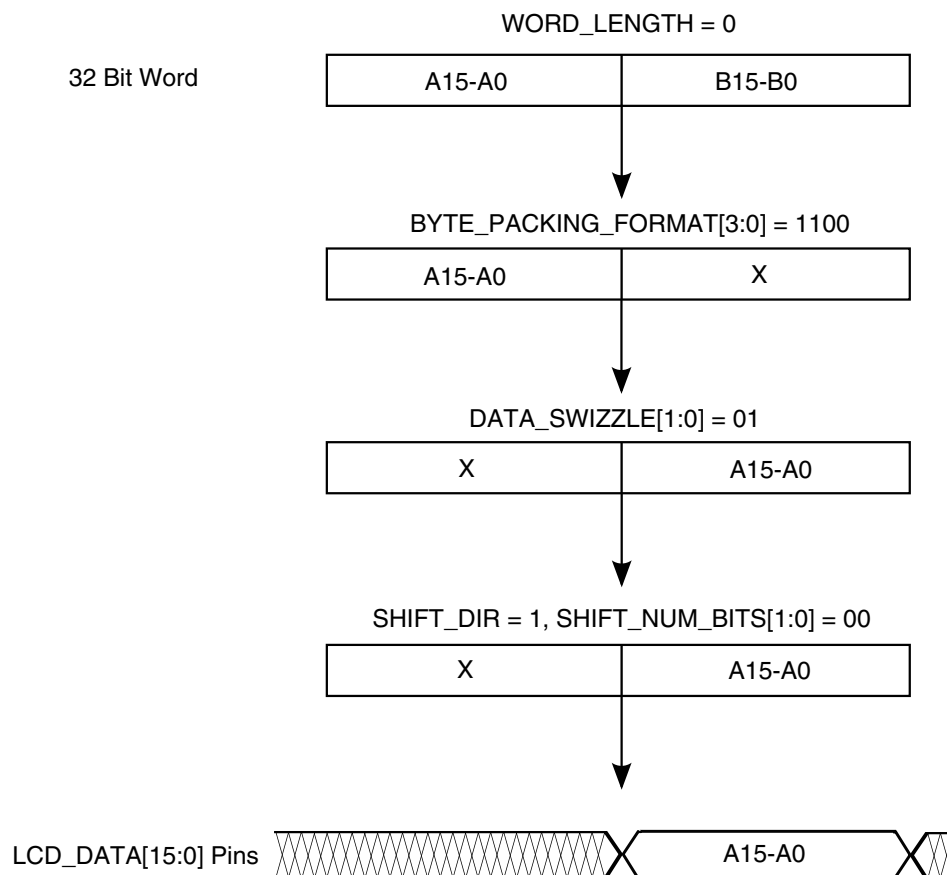


Figure 21-5. Register programming for write mode

This example indicates how an unpacked frame buffer can be sourced for display. Only a single 16 bit half word within the 32 bit word is transferred out via the 16 display bus.





### Figure 21-6. Register programming for write mode

### 21.4.3 Read Data Path

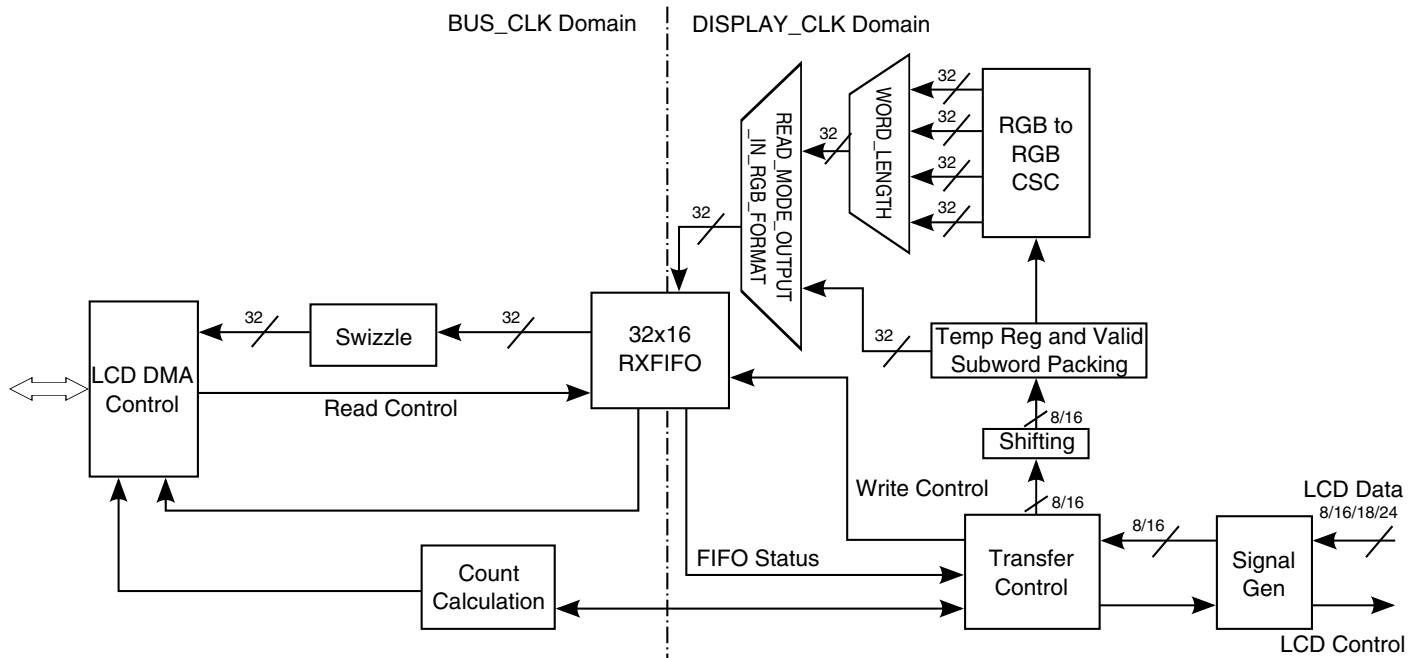
Figure 21-7 shows the MPU read data path in detail.

eLCDIF can read from an external display that follows the 6800/8080 MPU protocol.

The display bus width is determined by the LCD\_DATABUS\_WIDTH bit field. The data sampled at every read strobe is called a subword and the number of subwords that can be packed in a 32-bit word is given by the READ\_MODE\_NUM\_PACKED\_SUBWORDS bit field. The INITIAL\_DUMMY\_READ bit field directs the eLCDIF to skip the number of programmed subwords before starting to process read data. This feature is useful in the case of an LCD controller that returns the last written data the first time a read is issued, and then sends the correct data thereafter. SHIFT\_DIR and SHIFT\_NUM\_BITS bit fields indicate whether the data needs to be shifted before getting stored in the internal registers. For example, a value of 2 in READ\_MODE\_NUM\_PACKED\_SUBWORDS if lcd

## Functional Description

databus width is 8 bits indicates two bytes should be packed in a 32-bit word, while if the lcd databus width is 16 bits, it indicates that two half words (or 4 bytes) should be packed.



**Figure 21-7. MPU Read Data Path**

After the last subword within a word is reached, the block looks at the READ\_PACK\_DIR in the HW\_LCDIF\_CTRL2 register. If this bit is set, the block will swizzle the data, but only within the valid bytes, unlike in the write mode, where swizzle occurs across all 4 bytes. If the READ\_MODE\_OUTPUT\_IN\_RGB\_FORMAT bit is set, eLCDIF will convert the data obtained from the READ\_PACK\_DIR operation into 24-bit unpacked RGB and then re-convert it into 16/18/24 bpp RGB depending on the WORD\_LENGTH field. The DATA\_FORMAT\_16/18/24\_BIT bit fields are also considered while converting to 24-bit unpacked RGB format. For example, if DATA\_FORMAT\_18\_BIT is 1, the RGB666 data will be packed in the upper bits [31:4] of a 32-bit word, and that bit is 0, the data will be packed in the lower bits [17:0]. After all these operations, the data gets written into the RXFIFO.

The following figures show some examples of how data is handled in different MPU read modes.

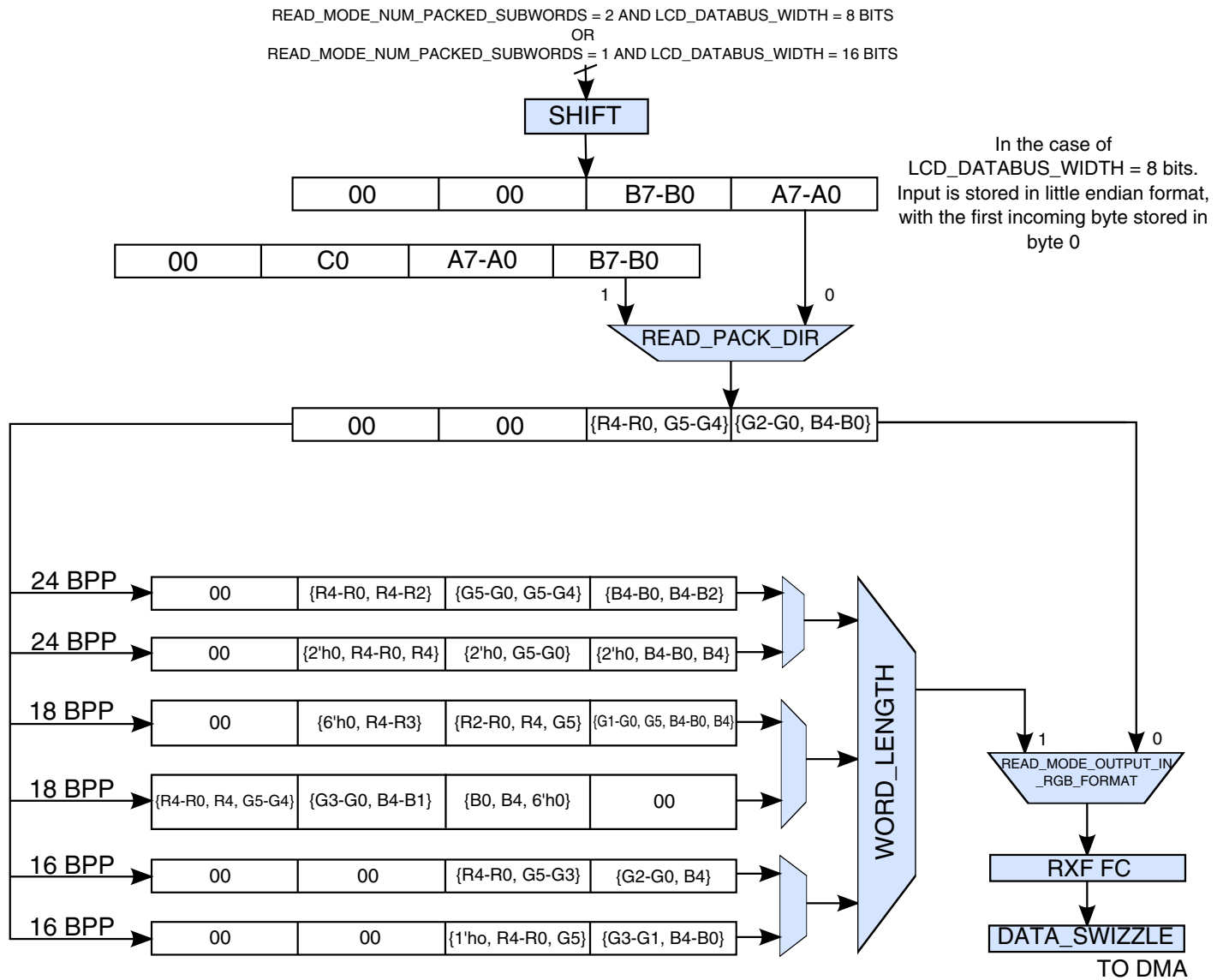
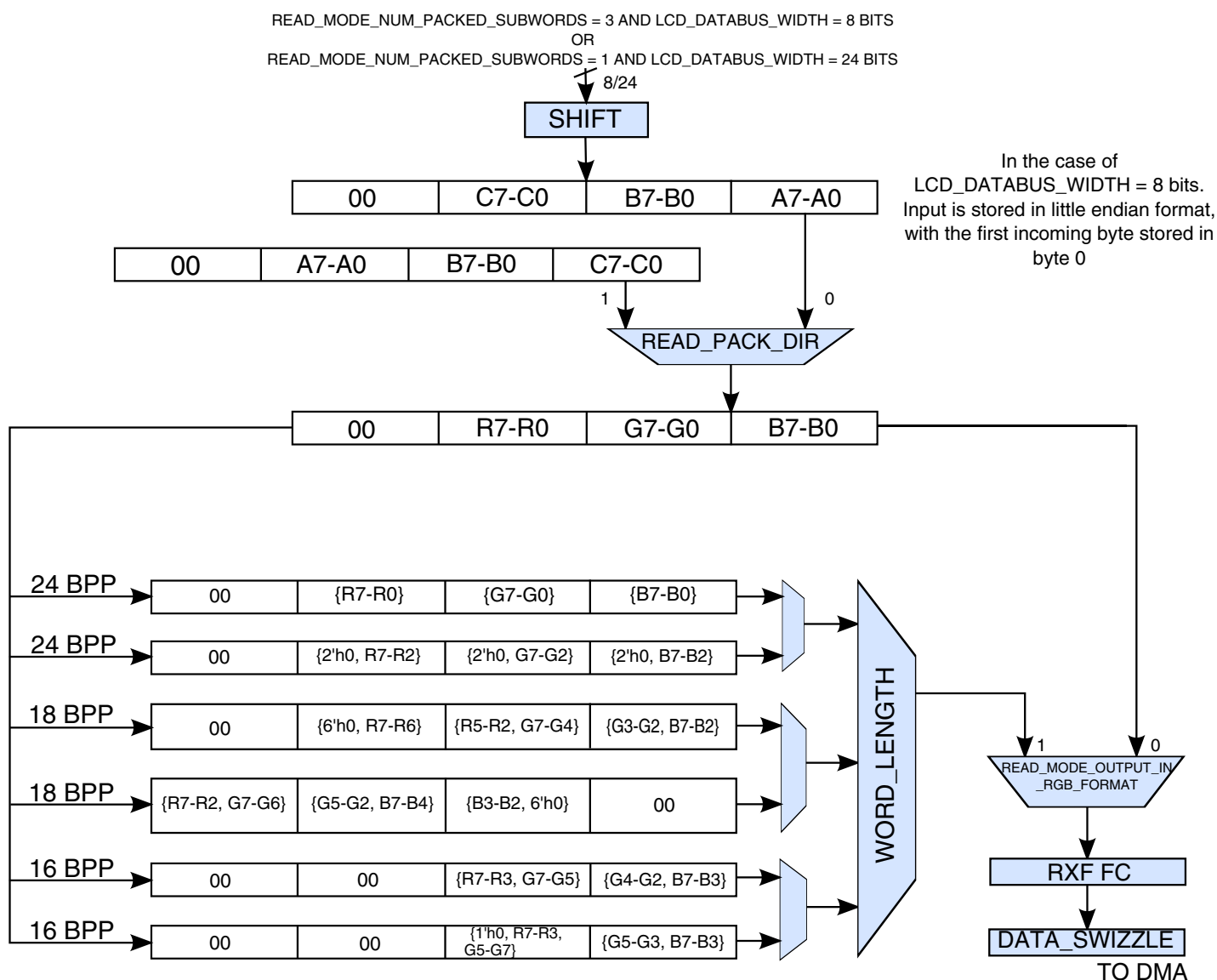


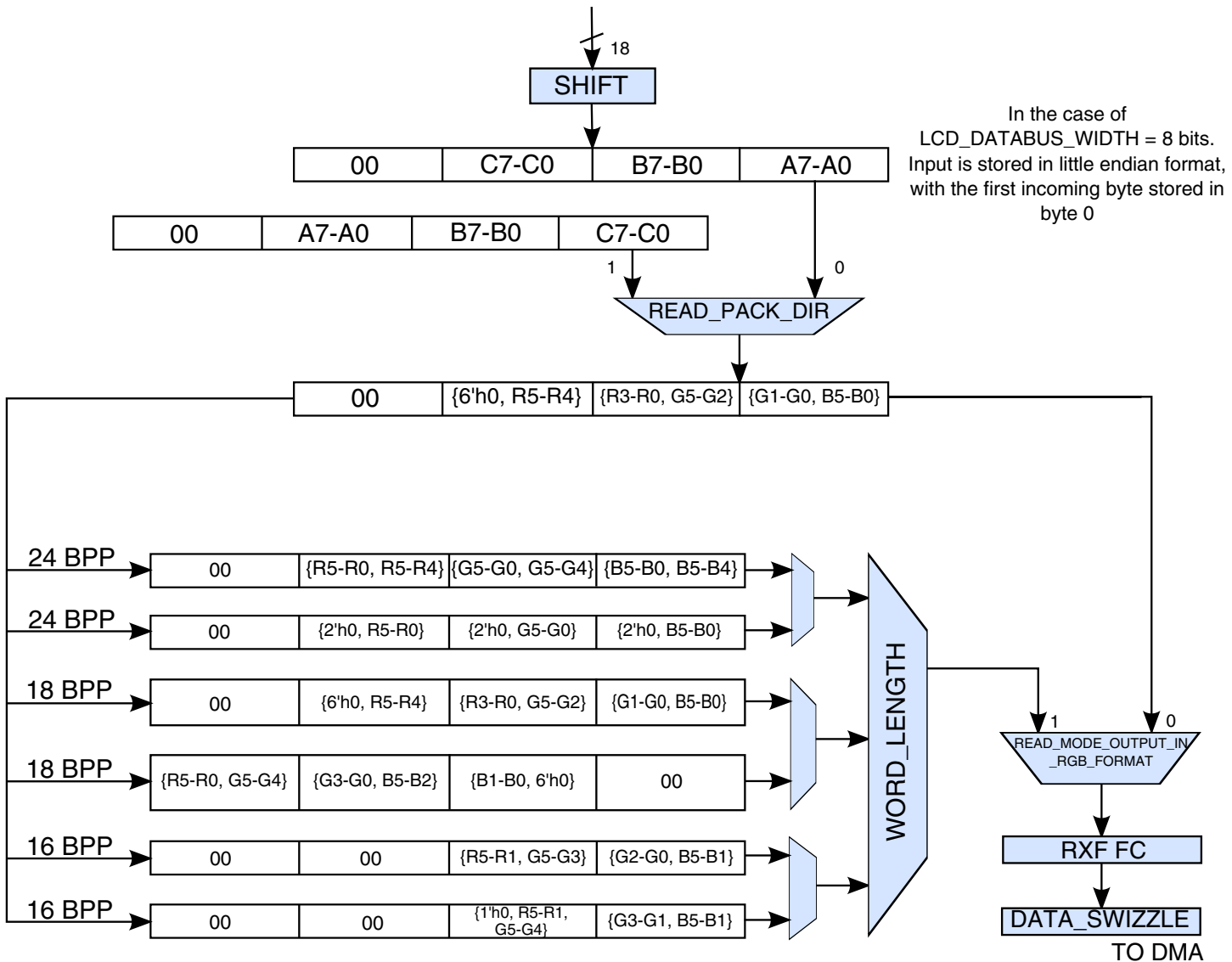
Figure 21-8. Data in MPU read mode

## Functional Description



**Figure 21-9. Data in MPU read mode**

READ\_MODE\_NUM\_PACKED\_SUBWORDS = 1 AND LCD\_DATABUS\_WIDTH = 18 BITS



**Figure 21-10. Data in MPU read mode**

#### Restrictions:

READ\_PACK\_DIR should only be used if it is required to swizzle the subwords before doing RGB to RGB CSC, otherwise the DATA\_SWIZZLE field should be used to swizzle across bytes.

READ\_PACK\_DIR must be 0 if LCD\_DATABUS\_WIDTH is 8 bits and READ\_MODE\_NUM\_PACKED\_SUBWORDS = 1

If READ\_MODE\_OUTPUT\_IN\_RGB\_FORMAT bit is set, the following restrictions should be followed:

- If LCD\_DATABUS\_WIDTH = 8 bits, then  
READ\_MODE\_NUM\_PACKED\_SUBWORDS <= 3.
- If LCD\_DATABUS\_WIDTH = 16/18/24 bits, then  
READ\_MODE\_NUM\_PACKED\_SUBWORDS = 1.

## 21.4.4 eLCDIF Interrupts

eLCDIF supports a number of interrupts to aid controlling and status reporting of the block.

All the interrupts have individual mask bits for enabling or disabling each of them. They all get funneled through a single interrupt line connected to the interrupt collector (ICOLL).

The following list describes the different interrupts supported by eLCDIF:

- Underflow interrupt is asserted when the clock domain crossing FIFO (TXFIFO) becomes empty but the block is in active display portion during that time. Software should take corrective action to make sure that this does not happen.
- In the bus master mode, the overflow interrupt will be asserted if the block has requested more data than its FIFOs could hold. In the read mode, it will be asserted if the RxFIFO becomes full and the block reads more data.
- VSYNC edge interrupt will be asserted every time a leading VSYNC edge occurs.
- Cur\_frame\_done interrupt occurs at the end of every frame in all modes except DVI. In DVI mode, if IRQ\_ON\_ALTERNATE\_FIELDS bit is set, it will occur at the end of every frame, otherwise it will occur at the end of every field.

## 21.4.5 Initializing the eLCDIF

This section describes write modes and MPU read mode.

### 21.4.5.1 Write Modes

The following initialization steps are common to all eLCDIF write modes of operation before entering any particular mode.

Initialization steps:

1. Configure the external I/Os to correctly interface the external display.
2. Start the DISPLAY\_CLK clock and set the appropriate frequency by programming the registers in CCM.

3. Start the BUS\_CLK and set the appropriate frequency by programming the registers in CCM.
4. Bring the eLCDIF out of soft reset and disable the clock gate bit.
5. Reset the LCD controller by setting LCDIF\_CTRL1\_RESET bit appropriately, being careful to observe the reset requirements of the controller. See [Behavior During Reset](#) for more information on Reset requirements.
6. Make sure READ\_WRITEB bit in HW\_LCDIF\_CTRL register is 0.
7. Select the transfer mode of operation. The LCDIF\_MASTER bit in HW\_LCDIF\_CTRL register determines the transfer mode selected. Bus master (LCDIF\_MASTER =1) or PIO (LCDIF\_MASTER =0) mode are the transfer modes to select.
8. Set the INPUT\_DATA\_SWIZZLE according to the endianness of the LCD controller. Also, set the DATA\_SHIFT\_DIR and SHIFT\_NUM\_BITS if it is required to shift the data left or right before it is output.
9. Set the WORD\_LENGTH field appropriately: 0 = 16-bit input, 1 = 8-bit input, 2 = 18-bit input, 3 = 24/32-bit input. Also, select the correct 16/18/24 bit data format with the corresponding fields in HW\_LCDIF\_CTRL register.
10. Set the BYTE\_PACKING\_FORMAT field in HW\_LCDIF\_CTRL1 according to the input frame.
11. Set the LCD\_DATABUS\_WIDTH appropriately: 0 = 16-bit output, 1 = 8-bit output, 2 = 18-bit output, 3 = 24/32-bit output.
12. Enable the necessary IRQs.

### 21.4.5.2 MPU Read Mode

The following initialization steps should be done to enter the MPU read mode of operation:

Initialization steps:

1. Configure the external I/Os to correctly interface the external display.
2. Start the DISPLAY\_CLK and set the appropriate frequency by programming the registers in CCM.
3. Start the BUS\_CLK and set the appropriate frequency by programming the registers in CCM.
4. Bring the eLCDIF out of soft reset and clock gate.
5. Reset the LCD controller by setting LCDIF\_CTRL1\_RESET bit appropriately, being careful to observe the reset requirements of the controller.
6. Set the READ\_WRITEB bit in LCDIF\_CTRL register to 1.
7. Set the LCDIF\_MASTER bit in LCDIF\_CTRL register to 0. Bus master mode is not supported for reading data from the display.

8. Also, set the DATA\_SHIFT\_DIR and SHIFT\_NUM\_BITS if it is required to shift the data left or right before it is output.
9. Indicate if the read data needs to color-space-converted and stored in a different RGB format by setting the READ\_MODE\_OUTPUT\_IN\_RGB\_FORMAT field accordingly.
10. Set the WORD\_LENGTH field appropriately: 0 = 16-bit input, 1 = 8-bit input, 2 = 18-bit input, 3 = 24-bit input if READ\_MODE\_OUTPUT\_IN\_RGB\_FORMAT is required. Also, select the correct 16/18/24 bit data format with the corresponding fields in LCDIF\_CTRL register.
11. Set the READ\_MODE\_NUM\_PACKED\_SUBWORDS field in LCDIF\_CTRL2 according to the number of subwords per word required to be packed.
12. Set the READ\_PACK\_DIR to 1 if it is required to store the data in big-endian format.
13. Set the LCD\_DATABUS\_WIDTH appropriately: 0 = 16-bit output, 1 = 8-bit output, 2 = 18-bit output, 3 = 24-bit output.
14. Enable the necessary IRQs.

## 21.4.6 MPU Interface

The MPU interface is used to transfer data and commands between the SoC via the eLCDIF and the external display at modest data rates.

Bus master or PIO transactions using the LCDIF\_DATA register can be used for MPU mode write operations. For MPU mode read operations, only PIO can be used. eLCDIF can support the 6800 as well as the 8080 MPU protocol. If DOTCLK\_MODE, DVI\_MODE and VSYNC\_MODE bits in LCDIF\_CTRL registers are 0, it implies that the block is in MPU interface mode of operation. The LCDIF MPU mode has four basic timing parameters: Setup and Hold for the Command/Data register selection (TCS, TCH) and Setup and Hold for the Data bus (TDS, TDH). These parameters are expressed in DISPLAY\_CLK cycles. The LCD\_WR signal is used as the write strobe while LCD\_RS signal is typically used to switch between command and data modes.



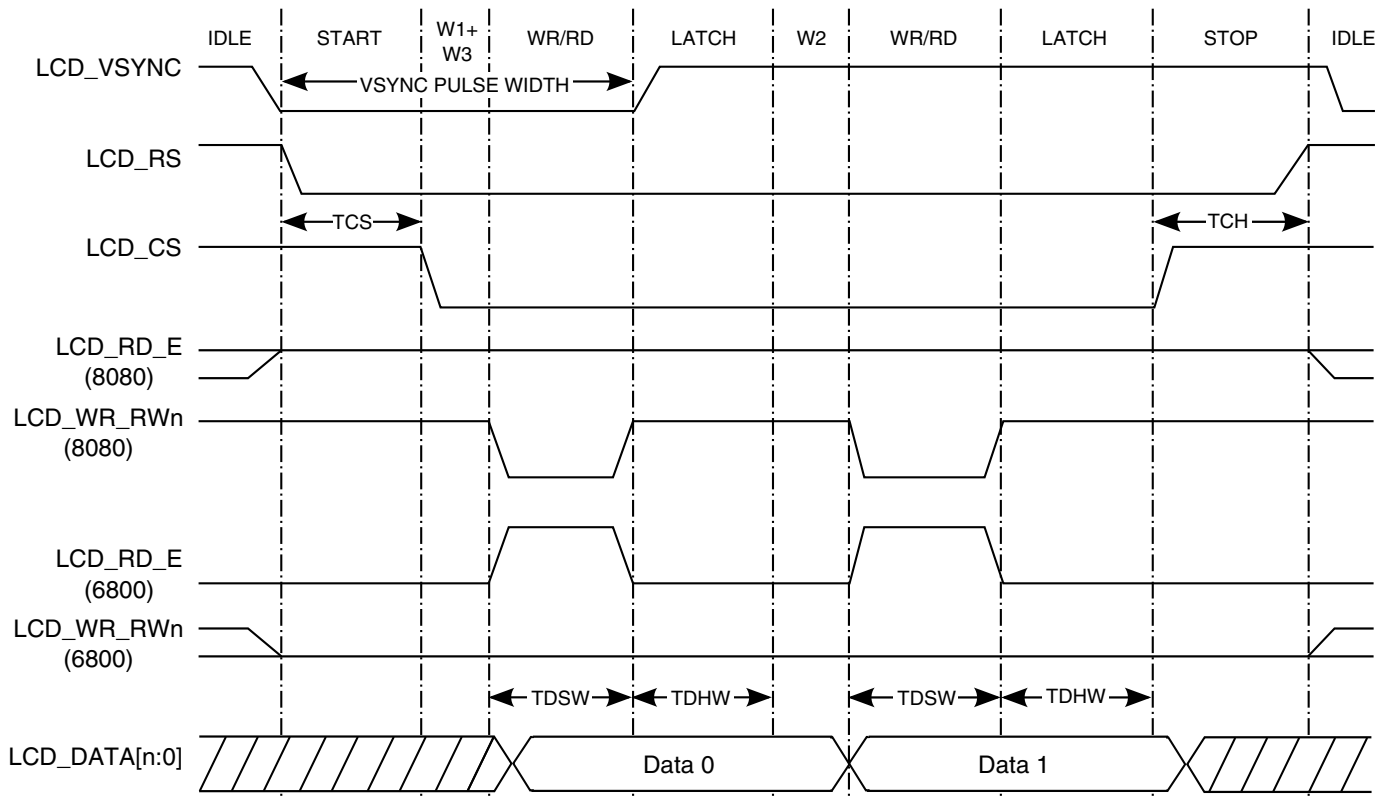


Figure 21-11. Timing in write mode of 6800 and 8080 protocols

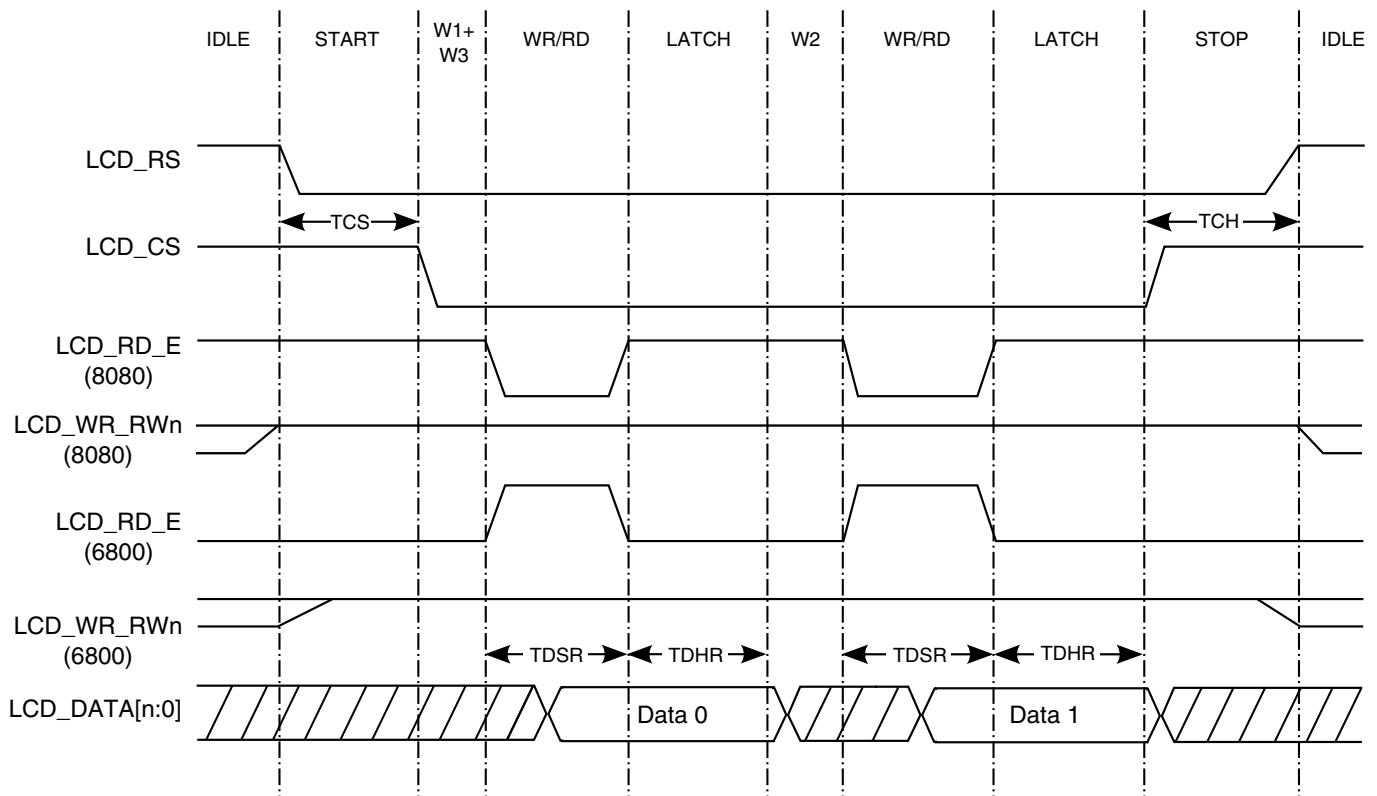


Figure 21-12. Read timing interface in 6800 and 8080 protocols

The eLCDIF has flexible pin and strobe timings which enable it to optimally support a wide range of LCDs. The minimum cycle time is two DISPLAY\_CLK cycles (TDS=TDH=1). For example, this results in a maximum LCD data rate of 12 MB/s when DISPLAY\_CLK is 24 MHz. TDS and TDH are 8-bit values, so the minimum eLCDIF period is 510 DISPLAY\_CLK cycles (47 KHz with a 24 MHz DISPLAY\_CLK). The timings are not automatically adjusted if the DISPLAY\_CLK frequency changes, so it may be necessary to adjust the timings if DISPLAY\_CLK changes.

In the MPU interface mode, the LCDIF\_CTRL\_BYPASS\_COUNT bit must be 0. The RUN bit is cleared automatically once the eLCDIF has received/transmitted all the data as per the LCDIF\_TRANSFER\_COUNT register and has completed the transfer to the panel. The current transfer can be cancelled/aborted if the RUN bit is manually made 0.

### 21.4.6.1 Code Example to Initialize the eLCDIF in MPU Write Mode

```
// Note: Common initialization steps in Initializing the eLCDIF must also be
// executed along with the following code
BF_CS1(LCDIF_CTRL, DATA_SELECT, 1); // 0 if sending command, 1 if sending data. Note that the
// idle state for LCD_RS signal is high, regardless of the
// programming of the DATA_SELECT register.

BF_CS1 (LCDIF_CTRL, MODE86, 8080_MODE);
BF_CS1 (LCDIF_CTRL, READ_WRITEB, 0);
BF_CS1 (LCDIF_CTRL, BYPASS_COUNT, 0); //Must be 0 in MPU mode
BF_CS1 (LCDIF_CTRL1, BUSY_ENABLE, 1); //Only if LCD controller implements a busy line
BF_CS4 (LCDIF_TIMING, CMD_HOLD, 2, CMD_SETUP, 2, DATA_HOLD, 2, DATA_SETUP, 2); //Values
based
// on DISPLAY_CLK frequency and timing requirements of
controller.
// Note that these register must be non-zero for correct
operation.
BF_CS2 (LCDIF_TRANSFER_COUNT, H_COUNT, 320, V_COUNT, 240); //For a 320 RGB x 240 display
BF_CS1 (LCDIF_CTRL, RUN, 1);
```

The eLCDIF is now ready to receive data via bus master PIO write transactions using the LCDIF\_DATA register. Note that when using the PIO write operations to the LCDIF\_DATA register, the software will need to poll the FIFO STATUS bits to ensure that it does not overflow the eLCDIF data buffers. When eLCDIF is done transmitting H\_COUNT x V\_COUNT pixels, it will stop, turn off the RUN bit and assert the cur\_frame\_done interrupt.

### 21.4.7 VSYNC Interface

The VSYNC interface uses the same protocol as the MPU interface, with an additional signal VSYNC at the frame rate of the display, as shown in the figure given in MPU Interface section.

It is used in the moving picture display mode where data has to be written to the internal LCD buffer at a speed higher than the display rate and displayed in synchronization with the VSYNC signal. This mode is selected by setting the VSYNC\_MODE bit in LCDIF\_CTRL register. The VSYNC signal is programmable for period, polarity and direction. Many other programmable parameters are shared with the MPU interface. The VSYNC\_OEB bit in LCDIF\_VDCTRL0 register indicates whether the display controller will send the VSYNC signal, or whether it should be generated by eLCDIF. The timing of the VSYNC signal is based on the DISPLAY\_CLK (make sure VSYNC\_PULSE\_WIDTH\_UNIT = VSYNC\_PERIOD\_UNIT = 0 and VSYNC\_ONLY = 1) and it is determined by the VSYNC\_PERIOD, VSYNC\_PULSE\_WIDTH and VSYNC\_POL fields in LCDIF\_VDCTRL0-4 registers. The SYNC\_SIGNALS\_ON bit in LCDIF\_VDCTRL4 register must be set if the target requires the VSYNC signal to be generated by eLCDIF. If the WAIT\_FOR\_VSYNC\_EDGE bit in LCDIF\_CTRL register is set, it indicates that the hardware should wait until it sees the leading VSYNC edge before starting the data transfer. The VERTICAL\_WAIT\_CNT indicates the number of DISPLAY\_CLK cycles from the leading VSYNC edge after which data transfer will be started on the interface.

In the VSYNC interface mode, the LCDIF\_CTRL\_BYPASS\_COUNT bit must be 0. The RUN bit is cleared automatically once the eLCDIF has received/transmitted all the data as per the LCDIF\_TRANSFER\_COUNT register and has completed the transfer to the panel. The current transfer can be cancelled/aborted if the RUN bit is manually made 0.

### 21.4.7.1 Code Example to Initialize eLCDIF in VSYNC Mode

```
// Note: Common initialization steps in Initializing the eLCDIF must also be
// executed along with the following code
BF_CS1 (LCDIF_CTRL, DATA_SELECT, 1); // 0 if sending command, 1 if sending data. Note that
//the idle state for LCD_RS signal is high, regardless of the programming of the DATA_SELECT
//register.

BF_CS1 (LCDIF_CTRL, MODE86, 8080_MODE);
BF_CS1 (LCDIF_CTRL, BYPASS_COUNT, 0); //Must be 0 in MPU mode
BF_CS1 (LCDIF_CTRL1, BUSY_ENABLE, 0);
BF_CS4 (LCDIF_TIMING, CMD_HOLD, 2, CMD_SETUP, 2, DATA_HOLD, 2, DATA_SETUP, 2); //Values
//based on DISPLAY_CLK frequency and timing requirements of controller. Note that these
//register must be non-zero for the MPU and VSYNC modes.
BF_CS2 (LCDIF_TRANSFER_COUNT, H_COUNT, 320, V_COUNT, 240); //For a 320 RGB x 240 display
//The following section indicates setting up the VSYNC signal timing when VSYNC is an output
BF_CS1 (LCDIF_VDCTRL0, VSYNC_OEB, 0); //Making VSYNC signal an output
BF_CS1 (LCDIF_VDCTRL4, VSYNC_ONLY, 1); //Only need to generate VSYNC signal
BF_CS1 (VDCTRL0, VSYNC_POL, 0); //Setting the polarity of VSYNC signal to be low during
//VSYNC PULSE WIDTH time
BF_CS2 (LCDIF_VDCTRL0, VSYNC_PERIOD_UNIT, 0, VSYNC_PULSE_WIDTH_UNIT, 0);
BF_CS2 (LCDIF_VDCTRL1, VSYNC_PERIOD, 400000, VSYNC_PULSE_WIDTH, 100); //Frame display rate in
//terms of number of DISPLAY_CLKs.
BF_CS2 (LCDIF_VDCTRL2, HSYNC_PULSE_WIDTH, 0, HSYNC_PERIOD, 0);
BF_CS1 (LCDIF_VDCTRL3, VERTICAL_WAIT_CNT, 50);
BF_CS1 (LCDIF_VDCTRL4, SYNC_SIGNALS_ON, 1);
```

## Functional Description

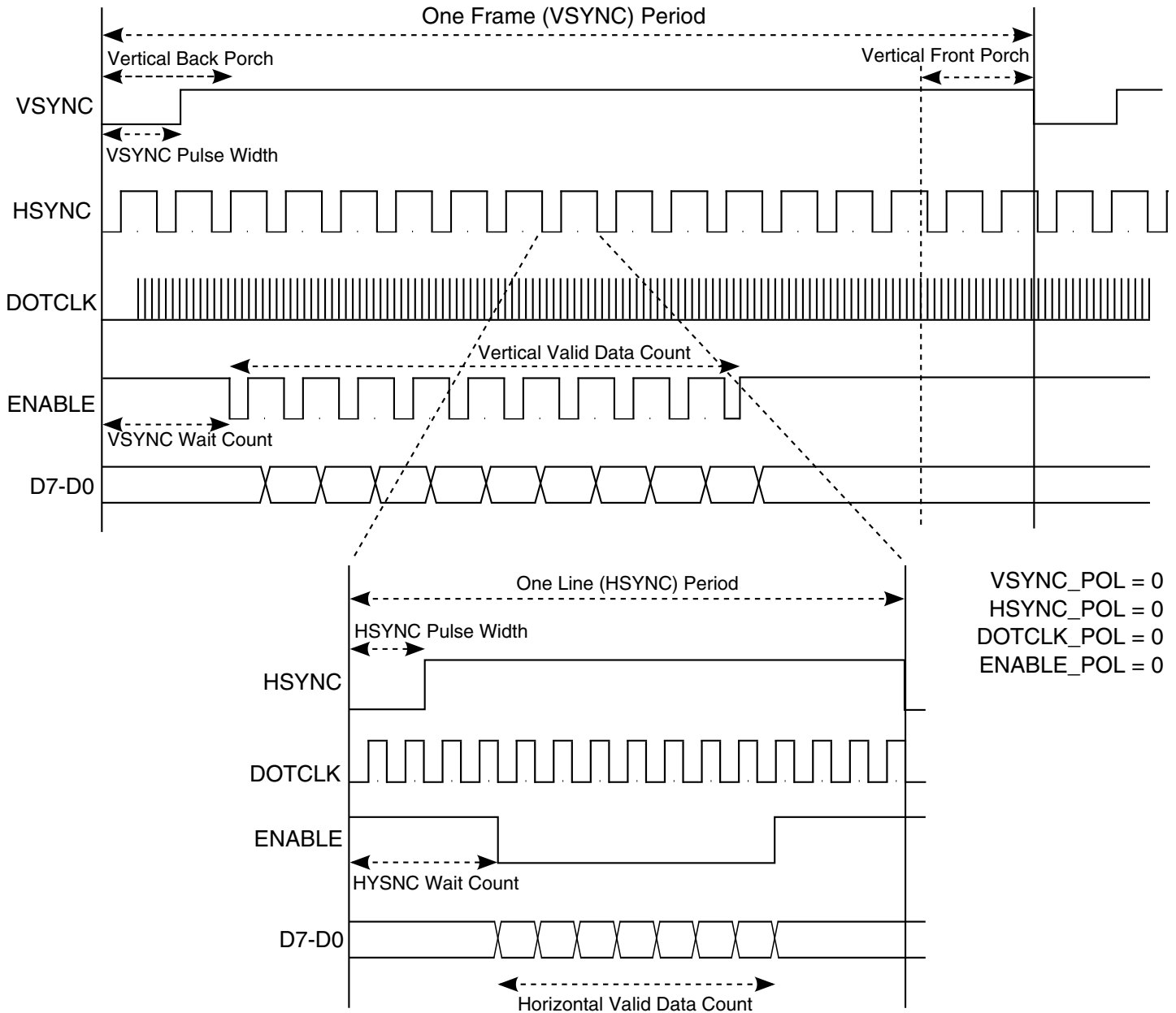
```
BF_CS2 (LCDIF_CTRL, VSYNC_MODE, 1, WAIT_FOR_VSYNC_EDGE, 1); //set WAIT_FOR_VSYNC_EDGE if
//software wishes to transfer the next frame after the VSYNC edge occurs.
BF_CS1 (LCDIF_CTRL, RUN, 1);
```

The eLCDIF is now ready to receive data via bus master requests or PIO writes to the LCDIF\_DATA register. When eLCDIF is done transmitting H\_COUNT x V\_COUNT pixels, it will stop, turn off the RUN bit and assert the cur\_frame\_done interrupt.

## 21.4.8 DOTCLK Interface

The DOTCLK interface is another mode used in moving picture displays.

It includes the VSYNC, HSYNC, DOTCLK and (optional) ENABLE signals. The interface is popularly called the RGB interface if the ENABLE signal is present.



**Figure 21-13. DOTCLK protocol with programmable parameters**

The DOTCLK mode writes data at high speed to the LCD, and the display operation is synchronized with the VSYNC, HSYNC, ENABLE and DOTCLK signals. The polarities, periods and pulse-widths of the sync signals are programmable using the LCDIF\_VDCTRL0-4 registers. The units for the VSYNC signal must be number of horizontal lines and can be selected using the VSYNC\_PULSE\_WIDTH\_UNIT and VSYNC\_PERIOD\_UNIT bit fields. The VERTICAL\_WAIT\_CNT is by default given the same unit as the VSYNC\_PERIOD. The DISPLAY\_CLK frequency is managed by the CCM.

In DOTCLK mode, LCDIF\_CTRL\_BYPASS\_COUNT bit must be set to 1. To end the current transfer, the software should make the DOTCLK\_MODE bit 0, so that all data that is currently in the LCDIF LFIFO and TXFIFO is transmitted. Once that transfer is complete, the block will automatically clear the RUN bit and issue the cur\_frame\_done interrupt.

### 21.4.8.1 Code Example

The following code shows an example for programming a 320x240 display. Note that setting up the display must be done through the MPU mode or via SPI.

```
// Note: Common initialization steps in Initializing the eLCDIF must also be
// executed along with the following code
BF_CS1 (LCDIF_CTRL, DOTCLK_MODE, 1);
BF_CS1 (LCDIF_CTRL, BYPASS_COUNT, 1); //Always for DOTCLK mode
BF_CS1 (LCDIF_VDCTRL0, VSYNC_OEB, 0); //Vsync is always an output in the DOTCLK mode
BF_CS4 (LCDIF_VDCTRL0, VSYNC_POL, 0, HSYNC_POL, 0, DOTCLK_POL, 0, ENABLE_POL, 0);
BF_CS1 (LCDIF_VDCTRL0, ENABLE_PRESENT, 1);
BF_CS2 (LCDIF_VDCTRL0, VSYNC_PERIOD_UNIT, 1, VSYNC_PULSE_WIDTH_UNIT, 1);
BF_CS1 (LCDIF_VDCTRL0, VSYNC_PULSE_WIDTH, 2);
BF_CS1 (LCDIF_VDCTRL1, VSYNC_PERIOD, 280);
BF_CS2 (LCDIF_VDCTRL2, HSYNC_PULSE_WIDTH, 10, HSYNC_PERIOD, 360); //Assuming
// LCD_DATABUS_WIDTH is 24bit
BF_CS2 (LCDIF_VDCTRL3, VSYNC_ONLY, 0);
BF_CS2 (LCDIF_VDCTRL3, HORIZONTAL_WAIT_CNT, 20, VERTICAL_WAIT_CNT, 20);
BF_CS1 (LCDIF_VDCTRL4, DOTCLK_H_VALID_DATA_CNT, 320); //Note that DOTCLK_V_VALID_DATA_CNT is
//implicitly assumed to be HW_LCDIF_TRANSFER_COUNT_V_COUNT
BF_CS1 (LCDIF_VDCTRL4, SYNC_SIGNALS_ON, 1);
BF_CS1 (LCDIF_CTRL, RUN, 1);
```

To stop the transfer completely, the ideal way is to make DOTCLK\_MODE = 0. In that case, the block will transmit the contents in the FIFO and reset the RUN bit.

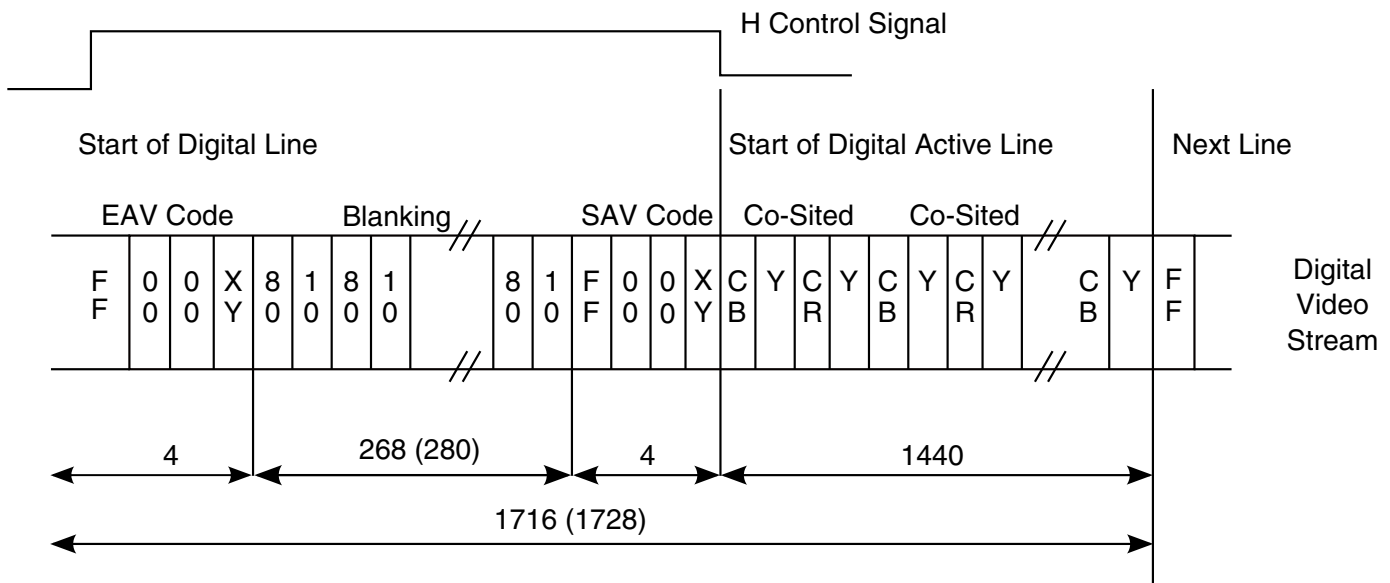
### 21.4.9 ITU-R BT.656 Digital Video Interface (DVI)

ITU-R BT.656 Digital Video Interface shown below transmits 4:2:2 YCbCr digital component video to a digital video encoder that can translate it into 525/60 or 625/50 analog TV signal.

Unique timing codes (timing reference signals) are embedded within the video stream to indicate the different timing events that would have been otherwise indicated by VSYNC, HSYNC and BLANK signals. The hardware supports 8-bit data transfers; the pins are shared with the lower 8 bits of LCD data bus. The LCD\_RS pin is shared with the clock signal of the interface (called CCIRCLK here for uniqueness). CCIRCLK also can be obtained on the LCD\_DOTCK pin. The mode shares the write FIFO with the LCD interface and the associated pipeline. The programmable parameters in registers LCDIF\_DVICTRL0-3 allow setting the total number of horizontal lines per frame, vertical and horizontal blanking interval, odd and even field start and end positions, and

so on. In short, these parameters are provided to ensure that the hardware has enough flexibility to generate the right 525/60 or 625/50 data streams. Most of the initialization steps in [Initializing the eLCDIF](#) such as data shifting, swizzle, and so on, are applicable to DVI mode also. The register descriptions in the programmable registers section at the end of this chapter include example code for programming the DVICTRL0-3 registers.

In DVI mode, LCDIF\_CTRL\_BYPASS\_COUNT bit must be set to 1. To end the current transfer, the software should make the DVI\_MODE bit the value 0, so that all data that is currently in the LCDIF LFIFO and TXFIFO is transmitted. Once that transfer is complete, the block will automatically clear the RUN bit and assert the cur\_frame\_done interrupt.



**Figure 21-14. Digital Video Interface**

### 21.4.10 eLCDIF Pin Usage by Interface Mode

[Table 21-3](#) and [Table 21-4](#) indicates how the eLCDIF level interface pins are used based on the desired mode of operation. The chip level I/Os should also be configured to be consistent with the desired eLCDIF operating mode.

The VSYNC signal has been mapped onto two pins, LCD\_BUSY and LCD\_VSYNC. The pin multiplexing can be programmed to select either of those pins to function as VSYN.

#### NOTE

There is an option to internally mux the HSYNC, DOTCLK and ENABLE signals in the DOTCLK mode by setting the MUX\_SYNC\_SIGNALS bit in the VDCTRL0 register. There

is also an option to internally mux the LCD\_WR\_RWn and LCD\_RD\_E pins in the CTRL1 register for backward compatibility.

**Table 21-3. Pin use in MPU Mode and VSYNC Mode**

PIN NAME	8-bit MPU LCD IF	16-bit MPU LCD IF	18-bit MPU LCD IF	24-bit MPU LCD IF	8-bit VSYNC LCD IF	16-bit VSYNC LCD IF	18-bit VSYNC LCD IF	24-bit VSYNC LCD IF
LCD_RS	LCD_RS	LCD_RS	LCD_RS	LCD_RS	LCD_RS	LCD_RS	LCD_RS	LCD_RS
LCD_CS	LCD_CS	LCD_CS	LCD_CS	LCD_CS	LCD_CS	LCD_CS	LCD_CS	LCD_CS
LCD_WR _RWn	LCD_WR _RWn	LCD_WR _RWn	LCD_WR _RWn	LCD_WR _RWn	LCD_WR _RWn	LCD_WR _RWn	LCD_WR _RWn	LCD_WR _RWn
LCD_RD_E	LCD_RD_E	LCD_RD_E	LCD_RD_E	LCD_RD_E	LCD_RD_E	LCD_RD_E	LCD_RD_E	LCD_RD_E
LCD_VSYNC * (Two options)	X	X	X	X	LCD_ VSYNC	LCD_ VSYNC	LCD_ VSYNC	LCD_ VSYNC
LCD_HSYNC	X	X	X	X	X	X	X	X
LCD_DOTCLK	X	X	X	X	X	X	X	X
LCD_ENABLE	X	X	X	X	X	X	X	X
LCD_D23	X	X	X	LCD_D23	X	X	X	LCD_D23
LCD_D22	X	X	X	LCD_D22	X	X	X	LCD_D22
LCD_D21	X	X	X	LCD_D21	X	X	X	LCD_D21
LCD_D20	X	X	X	LCD_D20	X	X	X	LCD_D20
LCD_D19	X	X	X	LCD_D19	X	X	X	LCD_D19
LCD_D18	X	X	X	LCD_D18	X	X	X	LCD_D18
LCD_D17	X	X	LCD_D17	LCD_D17	X	X	LCD_D17	LCD_D17
LCD_D16	X	X	LCD_D16	LCD_D16	X	X	LCD_D16	LCD_D16
LCD_D15 / VSYNC*	X	LCD_D15	LCD_D15	LCD_D15	VSYNC (optional)	LCD_D15	VSYNC (optional)	LCD_D15
LCD_D14 / HSYNC**	X	LCD_D14	LCD_D14	LCD_D14	X	LCD_D14	X	LCD_D14
LCD_D13 / LCD_DOTCLK**	X	LCD_D13	LCD_D13	LCD_D13	X	LCD_D13	X	LCD_D13
LCD_D12 / ENABLE**	X	LCD_D12	LCD_D12	LCD_D12	X	LCD_D12	X	LCD_D12
LCD_D11	X	LCD_D11	LCD_D11	LCD_D11	X	LCD_D11	X	LCD_D11
LCD_D10	X	LCD_D10	LCD_D10	LCD_D10	X	LCD_D10	X	LCD_D10
LCD_D9	X	LCD_D9	LCD_D9	LCD_D9	X	LCD_D9	X	LCD_D9
LCD_D8	X	LCD_D8	LCD_D8	LCD_D8	X	LCD_D8	X	LCD_D8

Table continues on the next page...



**Table 21-3. Pin use in MPU Mode and VSYNC Mode (continued)**

PIN NAME	8-bit MPU LCD IF	16-bit MPU LCD IF	18-bit MPU LCD IF	24-bit MPU LCD IF	8-bit VSYNC LCD IF	16-bit VSYNC LCD IF	18-bit VSYNC LCD IF	24-bit VSYNC LCD IF
LCD_D7	LCD_D7	LCD_D7	LCD_D7	LCD_D7	LCD_D7	LCD_D7	LCD_D7	LCD_D7
LCD_D6	LCD_D6	LCD_D6	LCD_D6	LCD_D6	LCD_D6	LCD_D6	LCD_D6	LCD_D6
LCD_D5	LCD_D5	LCD_D5	LCD_D5	LCD_D5	LCD_D5	LCD_D5	LCD_D5	LCD_D5
LCD_D4	LCD_D4	LCD_D4	LCD_D4	LCD_D4	LCD_D4	LCD_D4	LCD_D4	LCD_D4
LCD_D3	LCD_D3	LCD_D3	LCD_D3	LCD_D3	LCD_D3	LCD_D3	LCD_D3	LCD_D3
LCD_D2	LCD_D2	LCD_D2	LCD_D2	LCD_D2	LCD_D2	LCD_D2	LCD_D2	LCD_D2
LCD_D1	LCD_D1	LCD_D1	LCD_D1	LCD_D1	LCD_D1	LCD_D1	LCD_D1	LCD_D1
LCD_D0	LCD_D0	LCD_D0	LCD_D0	LCD_D0	LCD_D0	LCD_D0	LCD_D0	LCD_D0
LCD_RESET	LCD_RESE T	LCD_RESE T	LCD_RESE T	LCD_RESE T	LCD_RESE T	LCD_RESE T	LCD_RESE T	LCD_RESE T
LCD_BUSY / LCD_VSYNC	LCD_BUSY	LCD_BUSY	LCD_BUSY	LCD_BUSY	LCD_BUSY (OR optional LCD_VSYN C)	LCD_BUSY (OR optional LCD_VSYN C)	LCD_BUSY (OR optional LCD_VSYN C)	LCD_BUSY (OR optional LCD_VSYN C)

**Table 21-4. Pin use in DOTCLK Mode and DVI Mode**

PIN NAME	8-bit DOTCLK LCD IF	16-bit DOTCLK LCD IF	18-bit DOTCLK LCD IF	24-bit DOTCLK LCD IF	8-bit DVI LCD IF
LCD_RS	X	X	X	X	CCIR_CLK
LCD_CS	X	X	X	X	X
LCD_WR_RWn	X	X	X	X	X
LCD_RD_E	X	X	X	X	X
LCD_VSYNC* (Two options)	LCD_VSYNC	LCD_VSYNC	LCD_VSYNC	LCD_VSYNC	X
LCD_HSYNC	LCD_HSYNC	LCD_HSYNC	LCD_HSYNC	LCD_HSYNC	X
LCD_DOTCLK	LCD_DOTCLK	LCD_DOTCLK	LCD_DOTCLK	LCD_DOTCLK	X
LCD_ENABLE	LCD_ENABLE	LCD_ENABLE	LCD_ENABLE	LCD_ENABLE	X
LCD_D23	X	X	X	LCD_D23	X
LCD_D22	X	X	X	LCD_D22	X
LCD_D21	X	X	X	LCD_D21	X
LCD_D20	X	X	X	LCD_D20	X
LCD_D19	X	X	X	LCD_D19	X
LCD_D18	X	X	X	LCD_D18	X
LCD_D17	X	X	LCD_D17	LCD_D17	X
LCD_D16	X	X	LCD_D16	LCD_D16	X

Table continues on the next page...

**Table 21-4. Pin use in DOTCLK Mode and DVI Mode (continued)**

PIN NAME	8-bit DOTCLK LCD IF	16-bit DOTCLK LCD IF	18-bit DOTCLK LCD IF	24-bit DOTCLK LCD IF	8-bit DVI LCD IF
LCD_D15/ VSYNC*	X	LCD_D15	LCD_D15	LCD_D15	X
LCD_D14 / HSYNC**	X	LCD_D14	LCD_D14	LCD_D14	X
LCD_D13 / LCD_DOTCLK**	X	LCD_D13	LCD_D13	LCD_D13	X
LCD_D12 / ENABLE**	X	LCD_D12	LCD_D12	LCD_D12	X
LCD_D11	X	LCD_D11	LCD_D11	LCD_D11	X
LCD_D10	X	LCD_D10	LCD_D10	LCD_D10	X
LCD_D9	X	LCD_D9	LCD_D9	LCD_D9	X
LCD_D8	X	LCD_D8	LCD_D8	LCD_D8	X
LCD_D7	LCD_D7	LCD_D7	LCD_D7	LCD_D7	LCD_D7
LCD_D6	LCD_D6	LCD_D6	LCD_D6	LCD_D6	LCD_D6
LCD_D5	LCD_D5	LCD_D5	LCD_D5	LCD_D5	LCD_D5
LCD_D4	LCD_D4	LCD_D4	LCD_D4	LCD_D4	LCD_D4
LCD_D3	LCD_D3	LCD_D3	LCD_D3	LCD_D3	LCD_D3
LCD_D2	LCD_D2	LCD_D2	LCD_D2	LCD_D2	LCD_D2
LCD_D1	LCD_D1	LCD_D1	LCD_D1	LCD_D1	LCD_D1
LCD_D0	LCD_D0	LCD_D0	LCD_D0	LCD_D0	LCD_D0
LCD_RESET	LCD_RESET	LCD_RESET	LCD_RESET	LCD_RESET	X
LCD_BUSY / LCD_VSYNC	LCD_BUSY (OR optional LCD_VSYNC)	LCD_BUSY (OR optional LCD_VSYNC)	LCD_BUSY (OR optional LCD_VSYNC)	LCD_BUSY (OR optional LCD_VSYNC)	X

## 21.5 Behavior During Reset

BUS\_CLK and DISPLAY\_CLK must be running before making any changes to SFTRST or CLKGATE bits.

A soft reset (SFTRST) can take multiple clock periods to complete, so do not set CLKGATE when setting SFTRST.

The reset process gates the clocks automatically.

## 21.6 ELCDIF Memory Map/Register Definition

### eLCDIF Hardware Register Format Summary

**LCDIF memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20F_8000	eLCDIF General Control Register (LCDIF_CTRL)	32	R/W	C000_0000h	<a href="#">21.6.1/865</a>
20F_8004	eLCDIF General Control Register (LCDIF_CTRL_SET)	32	R/W	C000_0000h	<a href="#">21.6.1/865</a>
20F_8008	eLCDIF General Control Register (LCDIF_CTRL_CLR)	32	R/W	C000_0000h	<a href="#">21.6.1/865</a>
20F_800C	eLCDIF General Control Register (LCDIF_CTRL_TOG)	32	R/W	C000_0000h	<a href="#">21.6.1/865</a>
20F_8010	eLCDIF General Control1 Register (LCDIF_CTRL1)	32	R/W	000F_0000h	<a href="#">21.6.2/868</a>
20F_8014	eLCDIF General Control1 Register (LCDIF_CTRL1_SET)	32	R/W	000F_0000h	<a href="#">21.6.2/868</a>
20F_8018	eLCDIF General Control1 Register (LCDIF_CTRL1_CLR)	32	R/W	000F_0000h	<a href="#">21.6.2/868</a>
20F_801C	eLCDIF General Control1 Register (LCDIF_CTRL1_TOG)	32	R/W	000F_0000h	<a href="#">21.6.2/868</a>
20F_8020	eLCDIF General Control2 Register (LCDIF_CTRL2)	32	R/W	0020_0000h	<a href="#">21.6.3/870</a>
20F_8024	eLCDIF General Control2 Register (LCDIF_CTRL2_SET)	32	R/W	0020_0000h	<a href="#">21.6.3/870</a>
20F_8028	eLCDIF General Control2 Register (LCDIF_CTRL2_CLR)	32	R/W	0020_0000h	<a href="#">21.6.3/870</a>
20F_802C	eLCDIF General Control2 Register (LCDIF_CTRL2_TOG)	32	R/W	0020_0000h	<a href="#">21.6.3/870</a>
20F_8030	eLCDIF Horizontal and Vertical Valid Data Count Register (LCDIF_TRANSFER_COUNT)	32	R/W	0001_0000h	<a href="#">21.6.4/873</a>
20F_8040	LCD Interface Current Buffer Address Register (LCDIF_CUR_BUF)	32	R/W	0000_0000h	<a href="#">21.6.5/873</a>
20F_8050	LCD Interface Next Buffer Address Register (LCDIF_NEXT_BUF)	32	R/W	0000_0000h	<a href="#">21.6.6/874</a>
20F_8060	LCD Interface Timing Register (LCDIF_TIMING)	32	R/W	0000_0000h	<a href="#">21.6.7/874</a>
20F_8070	eLCDIF VSYNC Mode and Dotclk Mode Control Register0 (LCDIF_VDCTRL0)	32	R/W	0000_0000h	<a href="#">21.6.8/875</a>
20F_8074	eLCDIF VSYNC Mode and Dotclk Mode Control Register0 (LCDIF_VDCTRL0_SET)	32	R/W	0000_0000h	<a href="#">21.6.8/875</a>
20F_8078	eLCDIF VSYNC Mode and Dotclk Mode Control Register0 (LCDIF_VDCTRL0_CLR)	32	R/W	0000_0000h	<a href="#">21.6.8/875</a>
20F_807C	eLCDIF VSYNC Mode and Dotclk Mode Control Register0 (LCDIF_VDCTRL0_TOG)	32	R/W	0000_0000h	<a href="#">21.6.8/875</a>
20F_8080	eLCDIF VSYNC Mode and Dotclk Mode Control Register1 (LCDIF_VDCTRL1)	32	R/W	0000_0000h	<a href="#">21.6.9/877</a>
20F_8090	LCDIF VSYNC Mode and Dotclk Mode Control Register2 (LCDIF_VDCTRL2)	32	R/W	0000_0000h	<a href="#">21.6.10/877</a>
20F_80A0	eLCDIF VSYNC Mode and Dotclk Mode Control Register3 (LCDIF_VDCTRL3)	32	R/W	0000_0000h	<a href="#">21.6.11/878</a>
20F_80B0	eLCDIF VSYNC Mode and Dotclk Mode Control Register4 (LCDIF_VDCTRL4)	32	R/W	0000_0000h	<a href="#">21.6.12/879</a>

*Table continues on the next page...*

## LCDIF memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20F_80C0	Digital Video Interface Control0 Register (LCDIF_DVICTRL0)	32	R/W	0000_0000h	<a href="#">21.6.13/880</a>
20F_80D0	Digital Video Interface Control1 Register (LCDIF_DVICTRL1)	32	R/W	0000_0000h	<a href="#">21.6.14/880</a>
20F_80E0	Digital Video Interface Control2 Register (LCDIF_DVICTRL2)	32	R/W	0000_0000h	<a href="#">21.6.15/881</a>
20F_80F0	Digital Video Interface Control3 Register (LCDIF_DVICTRL3)	32	R/W	0000_0000h	<a href="#">21.6.16/882</a>
20F_8100	Digital Video Interface Control4 Register (LCDIF_DVICTRL4)	32	R/W	0000_0000h	<a href="#">21.6.17/883</a>
20F_8110	RGB to YCbCr 4:2:2 CSC Coefficient0 Register (LCDIF_CSC_COEFF0)	32	R/W	0000_0000h	<a href="#">21.6.18/884</a>
20F_8120	RGB to YCbCr 4:2:2 CSC Coefficient1 Register (LCDIF_CSC_COEFF1)	32	R/W	0000_0000h	<a href="#">21.6.19/885</a>
20F_8130	RGB to YCbCr 4:2:2 CSC Coefficient2 Register (LCDIF_CSC_COEFF2)	32	R/W	0000_0000h	<a href="#">21.6.20/886</a>
20F_8140	RGB to YCbCr 4:2:2 CSC Coefficient3 Register (LCDIF_CSC_COEFF3)	32	R/W	0000_0000h	<a href="#">21.6.21/887</a>
20F_8150	RGB to YCbCr 4:2:2 CSC Coefficient4 Register (LCDIF_CSC_COEFF4)	32	R/W	0000_0000h	<a href="#">21.6.22/887</a>
20F_8160	RGB to YCbCr 4:2:2 CSC Offset Register (LCDIF_CSC_OFFSET)	32	R/W	0080_0010h	<a href="#">21.6.23/888</a>
20F_8170	RGB to YCbCr 4:2:2 CSC Limit Register (LCDIF_CSC_LIMIT)	32	R/W	00FF_00FFh	<a href="#">21.6.24/889</a>
20F_8180	LCD Interface Data Register (LCDIF_DATA)	32	R/W	0000_0000h	<a href="#">21.6.25/890</a>
20F_8190	Bus Master Error Status Register (LCDIF_BM_ERROR_STAT)	32	R/W	0000_0000h	<a href="#">21.6.26/890</a>
20F_81A0	CRC Status Register (LCDIF_CRC_STAT)	32	R/W	0000_0000h	<a href="#">21.6.27/891</a>
20F_81B0	LCD Interface Status Register (LCDIF_STAT)	32	R	9500_0000h	<a href="#">21.6.28/891</a>
20F_81C0	LCD Interface Version Register (LCDIF_VERSION)	32	R	0400_0000h	<a href="#">21.6.29/893</a>
20F_81D0	LCD Interface Debug0 Register (LCDIF_DEBUG0)	32	R	0E81_0000h	<a href="#">21.6.30/894</a>
20F_81E0	LCD Interface Debug1 Register (LCDIF_DEBUG1)	32	R	0000_0000h	<a href="#">21.6.31/896</a>
20F_81F0	LCD Interface Debug2 Register (LCDIF_DEBUG2)	32	R	0000_0000h	<a href="#">21.6.32/897</a>
20F_8200	eLCDIF Threshold Register (LCDIF_THRES)	32	R/W	0100_000Fh	<a href="#">21.6.33/897</a>

## 21.6.1 eLCDIF General Control Register (LCDIF\_CTRLn)

The LCD Interface Control Register provides overall control of the eLCDIF block. The eLCDIF Control Register provides a variety of control functions to the programmer. These functions allow the interface to be very flexible to work with a variety of LCD controllers, and to minimize overhead and increase performance of LCD programming. The register has been organized such that switching between the different LCD modes can be done with minimum PIO writes.

Address: 20F\_8000h base + 0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SFTRST	CLKGATE	YCBCR422_INPUT	READ_WRITEB	WAIT_FOR_VSYNC_EDGE	DATA_SHIFT_DIR	SHIFT_NUM_BITS					DVI_MODE	BYPASS_COUNT	VSYNC_MODE	DOTCLK_MODE	DATA_SELECT
W																
Reset	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	INPUT_DATA_SWIZZLE		CSC_DATA_SWIZZLE		LCD_DATABUS_WIDTH		WORD_LENGTH		RGB_TO_YCBCR422_CSC	ENABLE_PXP_HANDSHAKE	MASTER	Reserved	DATA_FORMAT_16_BIT	DATA_FORMAT_18_BIT	DATA_FORMAT_24_BIT	RUN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### LCDIF\_CTRLn field descriptions

Field	Description
31 SFTRST	This bit must be set to zero to enable normal operation of the eLCDIF. When set to one, it forces a block level reset.
30 CLKGATE	This bit must be set to zero for normal operation. When set to one it gates off the clocks to the block.
29 YCBCR422_INPUT	Zero implies input data is in RGB color space. One implies input data is in YCbCr 4:2:2 format, such that YCbCr are packed in a 32-bit word. It also means that there are 2 pixels in 4 bytes. If this bit is set, software should program the H_COUNT field in the TRANSFER_COUNT register to the total number of pixels that will have to be fetched by the eLCDIF block per line and the BYTE_PACKING_FORMAT should be 0xF. The WORD_LENGTH does not matter in this case.

Table continues on the next page...

## LCDIF\_CTRLn field descriptions (continued)

Field	Description
28 READ_WRITEB	By default, eLCDIF is in the write mode. Setting this bit to 1 will make the hardware go into 6800/8080 MPU read mode. The LCDIF_MASTER bit must be 0, since bus master mode can only be used for writing the display.
27 WAIT_FOR_VSYNC_EDGE	Setting this bit to 1 will make the hardware wait for the triggering VSYNC edge before starting write transfers to the LCD. Used only in the VSYNC mode of operation.
26 DATA_SHIFT_DIR	Use this bit to determine the direction of shift of transmit data. In the DVI mode, it works only on the active data, not on the timing codes and ancillary data.  0x0 <b>TXDATA_SHIFT_LEFT</b> — Data to be transmitted is shifted LEFT by SHIFT_NUM_BITS bits. 0x1 <b>TXDATA_SHIFT_RIGHT</b> — Data to be transmitted is shifted RIGHT by SHIFT_NUM_BITS bits.
25–21 SHIFT_NUM_BITS	The data to be transmitted is shifted left or right by this number of bits.
20 DVI_MODE	Set this bit to 1 to get into the ITU-R BT.656 digital video interface mode. Toggle this bit from 1 to 0 to make the hardware go out of DVI mode after completing all data transfer and after the RUN bit has been deasserted.
19 BYPASS_COUNT	When this bit is 0, it means that eLCDIF will stop the block operation and turn off the RUN bit after the amount of data indicated by the LCDIF_TRANSFER_COUNT register has been transferred out. When this bit is set to 1, the block will continue normal operation indefinitely until it is told to stop. This bit must be 0 in MPU and VSYNC modes, and must be 1 in DOTCLK and DVI modes of operation.
18 VSYNC_MODE	Setting this bit to 1 will make the eLCDIF hardware go into VSYNC mode. WAIT_FOR_VSYNC_EDGE can be used only if this bit is set. If VSYNC signal is required to be an output from the block, SYNC_SIGNALS_ON bit in LCDIF_VDCTRL4 register must be set.
17 DOTCLK_MODE	Set this bit to 1 to make the hardware go into the DOTCLK mode, i.e. VSYNC/HSYNC/DOTCLK/ENABLE interface mode. ENABLE is optional, selected by the ENABLE_PRESENT bit. Toggle this bit from 1 to 0 to make the hardware go out of DOTCLK mode after completing all data transfer and deasserting the RUN bit.
16 DATA_SELECT	Command Mode polarity bit. This bit should only be changed when RUN is 0.  0x0 <b>CMD_MODE</b> — Command Mode. DCn signal is Low. 0x1 <b>DATA_MODE</b> — Data Mode. DCn signal is High.
15–14 INPUT_DATA_SWIZZLE	This field specifies how to swap the bytes fetched by the bus master interface. The swizzle function is independent of the WORD_LENGTH bit. The supported swizzle configurations are:  0x0 <b>NO_SWAP</b> — No byte swapping.(Little endian) 0x0 <b>LITTLE_ENDIAN</b> — Little Endian byte ordering (same as NO_SWAP). 0x1 <b>BIG_ENDIAN_SWAP</b> — Big Endian swap (swap bytes 0,3 and 1,2). 0x1 <b>SWAP_ALL_BYTES</b> — Swizzle all bytes, swap bytes 0,3 and 1,2 (aka Big Endian). 0x2 <b>HWD_SWAP</b> — Swap half-words. 0x3 <b>HWD_BYTE_SWAP</b> — Swap bytes within each half-word.
13–12 CSC_DATA_SWIZZLE	This field specifies how to swap the bytes after the data has been converted into an internal representation of 24 bits per pixel and before it is transmitted over the LCD interface bus. The data is always transmitted with the least significant byte/hword (half word) first after the swizzle takes place. So, INPUT_DATA_SWIZZLE takes place first on the incoming data, and then CSC_DATA_SWIZZLE is applied. The swizzle function is independent of the WORD_LENGTH or the LCD_DATABUS_WIDTH fields. If RGB_TO_YCRCB422_CSC bit is set, the swizzle occurs on the Y, Cb, Cr values. The supported swizzle configurations are:  0x0 <b>NO_SWAP</b> — No byte swapping.(Little endian)

Table continues on the next page...

## LCDIF\_CTRLn field descriptions (continued)

Field	Description
	0x0 <b>LITTLE_ENDIAN</b> — Little Endian byte ordering (same as NO_SWAP). 0x1 <b>BIG_ENDIAN_SWAP</b> — Big Endian swap (swap bytes 0,3 and 1,2). 0x1 <b>SWAP_ALL_BYTES</b> — Swizzle all bytes, swap bytes 0,3 and 1,2 (aka Big Endian). 0x2 <b>HWD_SWAP</b> — Swap half-words. 0x3 <b>HWD_BYTE_SWAP</b> — Swap bytes within each half-word.
11–10 LCD_DATABUS_WIDTH	LCD Data bus transfer width. 0x0 <b>16_BIT</b> — 16-bit data bus mode. 0x1 <b>8_BIT</b> — 8-bit data bus mode. 0x2 <b>18_BIT</b> — 18-bit data bus mode. 0x3 <b>24_BIT</b> — 24-bit data bus mode.
9–8 WORD_LENGTH	Input data format. 0x0 <b>16_BIT</b> — Input data is 16 bits per pixel. 0x1 <b>8_BIT</b> — Input data is 8 bits wide. 0x2 <b>18_BIT</b> — Input data is 18 bits per pixel. 0x3 <b>24_BIT</b> — Input data is 24 bits per pixel.
7 RGB_TO_YCBCR422_CSC	Set this bit to 1 to enable conversion from RGB to YCbCr colorspace. See the LCDIF_CSC_ registers for further details.
6 ENABLE_PXP_HANDSHAKE	If this bit is set and LCDIF_MASTER bit is set, the eLCDIF will act as bus master and the handshake mechanism between eLCDIF and ePXP will be turned on. If LCDIF_MASTER bit is not set, this bit becomes a don't care.
5 MASTER	Set this bit to make the eLCDIF act as a bus master. If this bit is reset, the eLCDIF will support or PIO mode.
4 RSRVD0	This field is reserved. Reserved bits. Write as 0.
3 DATA_FORMAT_16_BIT	When this bit is 1 and WORD_LENGTH = 0, it implies that the 16-bit data is in ARGB555 format. When this bit is 0 and WORD_LENGTH = 0, it implies that the 16-bit data is in RGB565 format. When WORD_LENGTH is not 0, this bit is a don't care.
2 DATA_FORMAT_18_BIT	Used only when WORD_LENGTH = 2, i.e. 18-bit. 0x0 <b>LOWER_18_BITS_VALID</b> — Data input to the block is in 18 bpp format, such that lower 18 bits contain RGB 666 and upper 14 bits do not contain any useful data. 0x1 <b>UPPER_18_BITS_VALID</b> — Data input to the block is in 18 bpp format, such that upper 18 bits contain RGB 666 and lower 14 bits do not contain any useful data.
1 DATA_FORMAT_24_BIT	Used only when WORD_LENGTH = 3, i.e. 24-bit. Note that this applies to both packed and unpacked 24-bit data. 0x0 <b>ALL_24_BITS_VALID</b> — Data input to the block is in 24 bpp format, such that all RGB 888 data is contained in 24 bits. 0x1 <b>DROP_UPPER_2_BITS_PER_BYTE</b> — Data input to the block is actually RGB 18 bpp, but there is 1 color per byte, hence the upper 2 bits in each byte do not contain any useful data, and should be dropped.
0 RUN	When this bit is set by software, the eLCDIF will begin transferring data between the SoC and the display. This bit must remain set until the operation is complete.

## 21.6.2 eLCDIF General Control1 Register (LCDIF\_CTRL1n)

The eLCDIF Control Register provides overall control of the eLCDIF block.

The eLCDIF Control1 Register provides additional programming to the eLCDIF. It implements some bits which are unlikely to change often in a particular application. It also carries interrupt-related bits which are common across more than one mode of operation.

Address: 20F\_8000h base + 10h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				COMBINE_MPU_WR_STRB	BM_ERROR_IRQ_EN	BM_ERROR_IRQ	RECOVER_ON_UNDERFLOW	INTERLACE_FIELDS	START_INTERLACE_FROM_SECOND_FIELD	FIFO_CLEAR	IRQ_ON_ALTERNATE_FIELDS	BYTE_PACKING_FORMAT			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OVERFLOW_IRQ_EN	UNDERFLOW_IRQ_EN	CUR_FRAME_DONE_IRQ_EN	VSYNC_EDGE_IRQ_EN	OVERFLOW_IRQ	UNDERFLOW_IRQ	CUR_FRAME_DONE_IRQ	VSYNC_EDGE_IRQ	Reserved				BUSY_ENABLE			MODE86
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**LCDIF\_CTRL1n field descriptions**

Field	Description
31–28 RSRVD1	This field is reserved. Reserved bits. Write as 0.
27 COMBINE_MPU_WR_STRB	If this bit is not set, the write strobe will be driven on LCD_WR_RWn pin in the 8080 mode and on the LCD_RD_E pin in the 6800 mode. If it is set, the write strobe of both the 6800 and 8080 modes will be driven only on the LCD_WR_RWn pin. Note that this does not work for read strobe.
26 BM_ERROR_IRQ_EN	This bit is set to enable bus master error interrupt in the eLCDIF master mode.
25 BM_ERROR_IRQ	This bit is set to indicate that an interrupt is requested by the eLCDIF block. This bit is cleared by software by writing a one to its SCT clear address. This bit will be set when the eLCDIF is in master mode and an error response was returned by the slave.

*Table continues on the next page...*



## LCDIF\_CTRL1n field descriptions (continued)

Field	Description
	0x0 <b>NO_REQUEST</b> — No Interrupt Request Pending. 0x1 <b>REQUEST</b> — Interrupt Request Pending.
24 RECOVER_ON_ UNDERFLOW	Set this bit to enable the eLCDIF block to recover in the next field/frame if there was an underflow in the current field/frame.
23 INTERLACE_ FIELDS	Set this bit if it is required that the eLCDIF block fetches odd lines in one field and even lines in the other field. It will work only in LCDIF_MASTER is set to 1.
22 START_ INTERLACE_ FROM_ SECOND_FIELD	The default is to grab the odd lines first and then the even lines. Set this bit if it is required to grab the even lines first and then the odd lines. (Line numbers start from 1, so odd lines are 1,3,5,etc. and even lines are 2,4,6, etc.)
21 FIFO_CLEAR	Set this bit to clear all the data in the latency FIFO (LFIFO), TXFIFO and the RXFIFO.
20 IRQ_ON_ ALTERNATE_ FIELDS	If this bit is set, the eLCDIF block will assert the cur_frame_done interrupt only on alternate fields, otherwise it will issue the interrupt on both odd and even field. This bit is mostly relevant if INTERLACE_FIELDS is set. This feature is only available in DOTCLK and DVI modes.
19–16 BYTE_ PACKING_ FORMAT	This bitfield is used to show which data bytes in a 32-bit word are valid. Default value 0xf indicates that all bytes are valid. For 8-bit transfers, any combination in this bitfield will mean valid data is present in the corresponding bytes. In the 16-bit mode, a 16-bit half-word is valid only if adjacent bits [1:0] or [3:2] or both are 1. A value of 0x0 will mean that none of the bytes are valid and should not be used. For example, set the bit field value to 0x7 if the display data is arranged in the 24-bit unpacked format (A-R-G-B where A value does not have to be transmitted). When input data is in YCbCr 4:2:2 format (YCBCR422_INPUT is 1), H_COUNT should be the number of pixels that should be fetched by the block and the BYTE_PACKING_FORMAT should be 0xF. (Note - YCBCR422_INPUT = 1 implies 2 pixels per 32 bits).
15 OVERFLOW_ IRQ_EN	This bit is set to enable an overflow interrupt in the TXFIFO in the write mode.
14 UNDERFLOW_ IRQ_EN	This bit is set to enable an underflow interrupt in the TXFIFO in the write mode.
13 CUR_FRAME_ DONE_IRQ_EN	This bit is set to 1 enable an interrupt every time the hardware enters in the vertical blanking state.
12 VSYNC_EDGE_ IRQ_EN	This bit is set to enable an interrupt every time the hardware encounters the leading VSYNC edge in the VSYNC and DOTCLK modes, or the beginning of every field in DVI mode.
11 OVERFLOW_ IRQ	This bit is set to indicate that an interrupt is requested by the eLCDIF block. This bit is cleared by software by writing a one to its SCT clear address. A latency FIFO (LFIFO) overflow in the write mode (MPU/VSYNC/DOTCLK/DVI mode) was detected, data samples have been lost.  0x0 <b>NO_REQUEST</b> — No Interrupt Request Pending. 0x1 <b>REQUEST</b> — Interrupt Request Pending.
10 UNDERFLOW_ IRQ	This bit is set to indicate that an interrupt is requested by the eLCDIF block. This bit is cleared by software by writing a one to its SCT clear address. A TXFIFO underflow in the write mode (MPU/VSYNC/DOTCLK/DVI mode) was detected. Could produce an error in the DOTCLK / DVI modes.

Table continues on the next page...

## LCDIF\_CTRL1n field descriptions (continued)

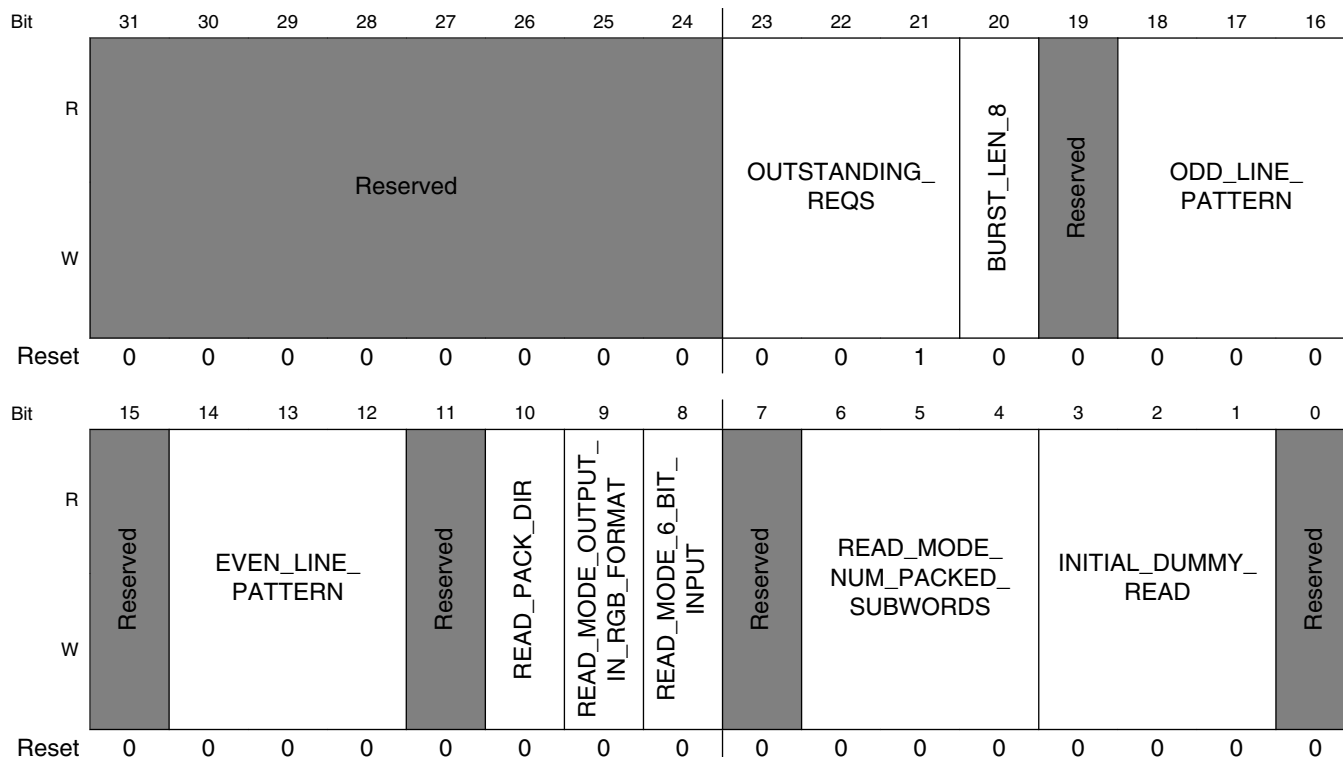
Field	Description
	0x0 <b>NO_REQUEST</b> — No Interrupt Request Pending. 0x1 <b>REQUEST</b> — Interrupt Request Pending.
9 CUR_FRAME_DONE_IRQ	This bit is set to indicate that an interrupt is requested by the eLCDIF block. This bit is cleared by software by writing a one to its SCT clear address. It indicates that the hardware has completed transmitting the current frame and is in the vertical blanking period in the DOTCLK/DVI modes. In the MPU and VSYNC modes, this IRQ is asserted at the end of the data transfer indicated by LCDIF_TRANSFER_COUNT register.  0x0 <b>NO_REQUEST</b> — No Interrupt Request Pending. 0x1 <b>REQUEST</b> — Interrupt Request Pending.
8 VSYNC_EDGE_IRQ	This bit is set to indicate that an interrupt is requested by the eLCDIF block. This bit is cleared by software by writing a one to its SCT clear address. It is set whenever the leading VSYNC edge is detected in the VSYNC and DOTCLK modes. In the DVI mode, it is asserted every time the block enters a new field.  0x0 <b>NO_REQUEST</b> — No Interrupt Request Pending. 0x1 <b>REQUEST</b> — Interrupt Request Pending.
7–3 RSRVD0	This field is reserved. Reserved bits. Write as 0.
2 BUSY_ENABLE	This bit enables the use of the interface's busy signal input. This should be enabled for LCD controllers that implement a busy line (to stall the eLCDIF from sending more data until ready). Otherwise this bit should be cleared.  0x0 <b>BUSY_DISABLED</b> — The busy signal from the LCD controller will be ignored. 0x1 <b>BUSY_ENABLED</b> — Enable the use of the busy signal from the LCD controller.
1 MODE86	This bit is used to select between the 8080 and 6800 series of microprocessor modes. This bit should only be changed when RUN is 0.  0x0 <b>8080_MODE</b> — Pins LCD_WR_RWn and LCD_RD_E function as active low WR and active low RD signals respectively. 0x1 <b>6800_MODE</b> — Pins LCD_WR_RWn and LCD_RD_E function as Read/Writeb and active high Enable signals respectively.
0 RESET	Reset bit for the external LCD controller. This bit can be changed at any time. It CANNOT be reset by SFTRST.  0x0 <b>LCDRESET_LOW</b> — LCD_RESET output signal is low. 0x1 <b>LCDRESET_HIGH</b> — LCD_RESET output signal is high.

### 21.6.3 eLCDIF General Control2 Register (LCDIF\_CTRL2n)

The eLCDIF Control Register provides overall control of the eLCDIF block.

The eLCDIF Control2 Register provides additional programming to the eLCDIF. It implements some bits which are unlikely to change often in a particular application.

Address: 20F\_8000h base + 20h offset + (4d × i), where i=0d to 3d



### LCDIF\_CTRL2n field descriptions

Field	Description
31–24 RSRVD5	This field is reserved. Reserved bits. Write as 0.
23–21 OUTSTANDING_ REQS	This bitfield indicates the maximum number of outstanding transactions that eLCDIF should request when it is acting as a bus master. Default is 2 outstanding transactions.  0x0 <b>REQ_1</b> — 0x1 <b>REQ_2</b> — 0x2 <b>REQ_4</b> — 0x3 <b>REQ_8</b> — 0x4 <b>REQ_16</b> —
20 BURST_LEN_8	By default, when the eLCDIF is in the bus master mode, it will issue AXI bursts of length 16 (except when in packed 24 bpp mode, it will issue bursts of length 15). When this bit is set to 1, the block will issue bursts of length 8 (except when in packed 24 bpp mode, it will issue bursts of length 9). Note that this bitfield is only applicable when LCDIF_MASTER is set to 1.
19 RSRVD4	This field is reserved. Reserved bits. Write as 0.
18–16 ODD_LINE_ PATTERN	This field determines the order of the RGB components of each pixel in ODD lines (line numbers 1,3,5,...). This bitfield must be 0 in DVI mode.  0x0 <b>RGB</b> — 0x1 <b>RBG</b> — 0x2 <b>GBR</b> — 0x3 <b>GRB</b> —

Table continues on the next page...

## LCDIF\_CTRL2n field descriptions (continued)

Field	Description
	0x4 <b>BRG</b> — 0x5 <b>BGR</b> —
15 RSRVD3	This field is reserved. Reserved bits. Write as 0.
14–12 EVEN_LINE_ PATTERN	This field determines the order of the RGB components of each pixel in EVEN lines (line numbers 2,4,6,...). This bitfield must be 0 in DVI mode.  0x0 <b>RGB</b> — 0x1 <b>RBG</b> — 0x2 <b>GBR</b> — 0x3 <b>GRB</b> — 0x4 <b>BRG</b> — 0x5 <b>BGR</b> —
11 RSRVD2	This field is reserved. Reserved bits. Write as 0.
10 READ_PACK_DIR	The default value of 0 indicates data is stored in the little endian format. When LCD_DATABUS_WIDTH is 8-bit, this bit provides the option of rearranging the data byte-wise in the big endian format. For example, if READ_MODE_NUM_PACKED_SUBWORDS = 3 and the order of incoming data is 0x11, 0x22 and 0x33, then setting this bit to 1 will cause the data to be stored as 0x00112233 as opposed to the default 0x00332211. This operation occurs after the shifting operation done by SHIFT_NUM_BITS bitfield.
9 READ_MODE_ OUTPUT_IN_ RGB_FORMAT	Setting this bit will enable the eLCDIF to convert the incoming data to the RGB format given by WORD_LENGTH bitfield. This feature is not available when WORD_LENGTH is set to 8 bits. eLCDIF performs this operation of converting to RGB format after the endianness has been determined by the READ_PACK_DIR bitfield.
8 READ_MODE_6_ BIT_INPUT	Setting this bit to 1 indicates to eLCDIF that even though LCD_DATABUS_WIDTH is set to 8 bits, the input data is actually only 6 bits wide and exists on D5-D0.
7 RSRVD1	This field is reserved. Reserved bits. Write as 0.
6–4 READ_MODE_ NUM_PACKED_ SUBWORDS	Indicates the number of valid 8/16/18/24-bit subwords that will be packed into the 32-bit word in read mode. The subword size (8,16, 18 or 24 bits) is determined by the LCD_DATABUS_WIDTH field. The swizzle operation is performed after READ_MODE_NUM_PACKED_SUBWORDS number of subwords has been received and stored in little-endian format. For example, if LCD_DATABUS_WIDTH is set to 8-bit and data to be read back has to be stored in memory in 24-bit unpacked RGB format, set READ_MODE_NUM_PACKED_SUBWORDS to 0x3 so that each 32-bit word will contain only 3 valid bytes (RGB). Maximum value of READ_MODE_NUM_PACKED_SUBWORDS is 4 for 8-bit databus, 2 for 16-bit databus and 1 for 18/24-bit databus.
3–1 INITIAL_DUMMY_ READ	The value in this field determines the number of dummy 8/16/18/24-bit subwords that have to be read back from the LCD panel/controller. They will then not be stored in the read FIFO.
0 RSRVD0	This field is reserved. Reserved bits. Write as 0.

### 21.6.4 eLCDIF Horizontal and Vertical Valid Data Count Register (LCDIF\_TRANSFER\_COUNT)

This register tells the eLCDIF how much data will be sent for this frame, or transaction. The total number of words is a product of the V\_COUNT and H\_COUNT fields. The word size is specified by the WORD\_LENGTH field.

This register gives the dimensions of the input frame. For normal operation, but V\_COUNT and H\_COUNT should be non-zero.

Address: 20F\_8000h base + 30h offset = 20F\_8030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	V_COUNT																H_COUNT															
W	V_COUNT																H_COUNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### LCDIF\_TRANSFER\_COUNT field descriptions

Field	Description
31–16 V_COUNT	Number of horizontal lines per frame which contain valid data. In DOTCLK mode, V_COUNT should be the same as the number of active horizontal lines in a progressive frame. In DVI mode, V_COUNT should be the number of active horizontal lines per frame, and not per field.
15–0 H_COUNT	Total valid data (pixels) in each horizontal line. The data size is given by the WORD_LENGTH. When input data is in YCbCr 4:2:2 format (YCBCR422_INPUT is 1), H_COUNT should be the number of 32-bit words that should be fetched by the block and the BYTE_PACKING_FORMAT should be 0xF. In 24-bit packed format (WORD_LENGTH=0x3, BYTE_PACKING_FORMAT=0xF), the H_COUNT must be a multiple of 4 pixels. In 16-bit packed format (WORD_LENGTH=0x0, BYTE_PACKING_FORMAT=0xF), the H_COUNT must be a multiple of 2 pixels.

### 21.6.5 LCD Interface Current Buffer Address Register (LCDIF\_CUR\_BUF)

This register indicates the address of the current frame being transmitted by eLCDIF.

When the eLCDIF is behaving as a master, this address points to the address of the current frame of data being sent out via the LCDIF. When the current frame is done, the LCDIF block will assert the cur\_frame\_done interrupt for software to take action. The block will also copy the LCDIF\_NEXT\_BUF\_ADDR into this bitfield so that the software can program the next frame address into the LCDIF\_NEXT\_BUF\_ADDR bitfield. This address must always be double-word aligned.

## ELCDIF Memory Map/Register Definition

Address: 20F\_8000h base + 40h offset = 20F\_8040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### LCDIF\_CUR\_BUF field descriptions

Field	Description
31–0 ADDR	-

## 21.6.6 LCD Interface Next Buffer Address Register (LCDIF\_NEXT\_BUF)

This register indicates the address of next frame that will be transmitted by eLCDIF.

When the eLCDIF is behaving as a master, this address points to the address of the next frame of data that will be sent out via the eLCDIF. It is upto the software to make sure that this register is programmed before the end of the current frame, otherwise it might result in old data going out the eLCDIF. This address must always be double-word aligned.

Address: 20F\_8000h base + 50h offset = 20F\_8050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### LCDIF\_NEXT\_BUF field descriptions

Field	Description
31–0 ADDR	-

## 21.6.7 LCD Interface Timing Register (LCDIF\_TIMING)

The LCD interface timing register controls the various setup and hold times enforced by the LCD interface in the 6800/8080 MPU and VSYNC modes of operation.

The values used in this register are dependent on the particular LCD controller used, consult the users manual for the particular controller for required timings. Each field of the register must be non-zero, therefore the minimum value is: 0x01010101. NOTE: the timings are not automatically adjusted if the CLK\_DIS\_LCDIFn frequency changes--it

may be necessary to adjust the timings if CLK\_DIS\_LCDIFn changes. NOTE: Each field in this register must be non-zero for the MPU and VSYNC modes to function. The settings in this register do not affect the DOTCLK and DVI modes.

Address: 20F\_8000h base + 60h offset = 20F\_8060h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMD_HOLD								CMD_SETUP								DATA_HOLD								DATA_SETUP							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### LCDIF\_TIMING field descriptions

Field	Description
31–24 CMD_HOLD	Number of CLK_DIS_LCDIFn cycles that the DCn signal is active after CEn is deasserted.
23–16 CMD_SETUP	Number of CLK_DIS_LCDIFn cycles that the the DCn signal is active before CEn is asserted.
15–8 DATA_HOLD	Data bus hold time in CLK_DIS_LCDIFn cycles. Also the time that the data strobe is de-asserted in a cycle
7–0 DATA_SETUP	Data bus setup time in CLK_DIS_LCDIFn cycles. Also the time that the data strobe is asserted in a cycle.

## 21.6.8 eLCDIF VSYNC Mode and Dotclk Mode Control Register0 (LCDIF\_VDCTRL0n)

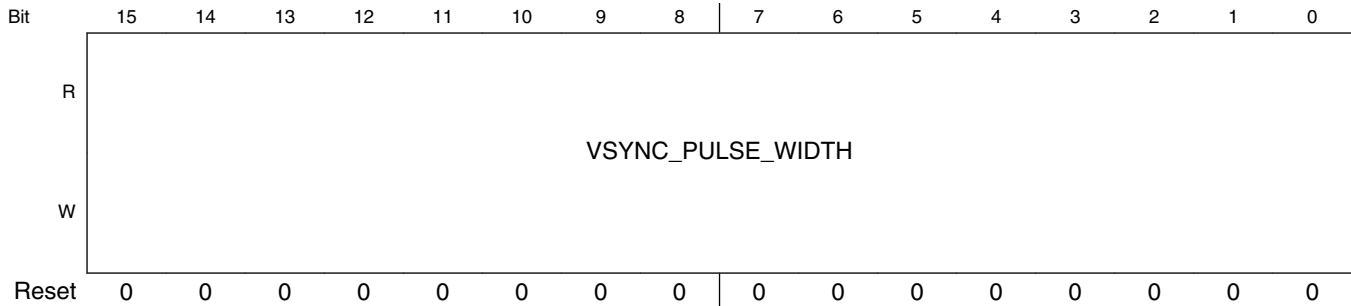
This register is used to control the VSYNC and DOTCLK modes of the LCDIF so as to work with different types of LCDs like moving picture displays and delta pixel displays.

This register gives general programmability to the VSYNC signal including polarity, direction, pulse width, etc.

Address: 20F\_8000h base + 70h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## ELCDIF Memory Map/Register Definition



### LCDIF\_VDCTRL0n field descriptions

Field	Description
31–30 RSRVD2	This field is reserved. Reserved bits. Write as 0.
29 VSYNC_OEB	0 means the VSYNC signal is an output, 1 means it is an input. Should be set to 0 in the DOTCLK mode.  0x0 <b>VSYNC_OUTPUT</b> — The VSYNC pin is in the output mode and the VSYNC signal has to be generated by the eLCDIF block. 0x1 <b>VSYNC_INPUT</b> — The VSYNC pin is in the input mode and the LCD controller sends the VSYNC signal to the block.
28 ENABLE_PRESENT	Setting this bit to 1 will make the hardware generate the ENABLE signal in the DOTCLK mode, thereby making it the true RGB interface along with the remaining three signals VSYNC, HSYNC and DOTCLK.
27 VSYNC_POL	Default 0 active low during VSYNC_PULSE_WIDTH time and will be high during the rest of the VSYNC period. Set it to 1 to invert the polarity.
26 HSYNC_POL	Default 0 active low during HSYNC_PULSE_WIDTH time and will be high during the rest of the HSYNC period. Set it to 1 to invert the polarity.
25 DOTCLK_POL	Default is data launched at negative edge of DOTCLK and captured at positive edge. Set it to 1 to invert the polarity. Set it to 0 in DVI mode.
24 ENABLE_POL	Default 0 active low during valid data transfer on each horizontal line.
23–22 RSRVD1	This field is reserved. Reserved bits. Write as 0.
21 VSYNC_PERIOD_UNIT	Default 0 for counting VSYNC_PERIOD in terms of CLK_DIS_LCDIFn cycles. Set it to 1 to count in terms of complete horizontal lines. CLK_DIS_LCDIFn cycles should be used in the VSYNC mode, while horizontal line should be used in the DOTCLK mode.
20 VSYNC_PULSE_WIDTH_UNIT	Default 0 for counting VSYNC_PULSE_WIDTH in terms of CLK_DIS_LCDIFn cycles. Set it to 1 to count in terms of complete horizontal lines.
19 HALF_LINE	Setting this bit to 1 will make the total VSYNC period equal to the VSYNC_PERIOD field plus half the HORIZONTAL_PERIOD field (i.e. VSYNC_PERIOD field plus half horizontal line), otherwise it is just VSYNC_PERIOD. Should be only used in the DOTCLK mode, not in the VSYNC interface mode.
18 HALF_LINE_MODE	When this bit is 0, the first field (VSYNC period) will end in half a horizontal line and the second field will begin with half a horizontal line. When this bit is 1, all fields will end with half a horizontal line, and none will begin with half a horizontal line.
17–0 VSYNC_PULSE_WIDTH	Number of units for which VSYNC signal is active. For the DOTCLK mode, the unit is determined by the VSYNC_PULSE_WIDTH_UNIT. If the VSYNC_PULSE_WIDTH_UNIT is 0 for DOTCLK mode, VSYNC_PULSE_WIDTH must be less than HSYNC_PERIOD. For the VSYNC interface mode, it should be in terms of number of CLK_DIS_LCDIFn cycles only.



### 21.6.9 eLCDIF VSYNC Mode and Dotclk Mode Control Register1 (LCDIF\_VDCTRL1)

This register is used to control the VSYNC signal in the VSYNC and DOTCLK modes of the block.

This register determines the period and duty cycle of the VSYNC signal when it is generated in the block.

Address: 20F\_8000h base + 80h offset = 20F\_8080h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VSYNC_PERIOD																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### LCDIF\_VDCTRL1 field descriptions

Field	Description
31–0 VSYNC_PERIOD	Total number of units between two positive or two negative edges of the VSYNC signal. If HALF_LINE is set, it is implicitly calculated to be VSYNC_PERIOD plus half HSYNC_PERIOD.

### 21.6.10 LCDIF VSYNC Mode and Dotclk Mode Control Register2 (LCDIF\_VDCTRL2)

This register is used to control the HSYNC signal in the DOTCLK mode of the block.

This register determines the period and duty cycle of the HSYNC signal when it is generated in the block.

Address: 20F\_8000h base + 90h offset = 20F\_8090h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HSYNC_PULSE_WIDTH																HSYNC_PERIOD															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### LCDIF\_VDCTRL2 field descriptions

Field	Description
31–18 HSYNC_PULSE_WIDTH	Number of CLK_DIS_LCDIFn cycles for which HSYNC signal is active.
17–0 HSYNC_PERIOD	Total number of CLK_DIS_LCDIFn cycles between two positive or two negative edges of the HSYNC signal.

## 21.6.11 eLCDIF VSYNC Mode and Dotclk Mode Control Register3 (LCDIF\_VDCTRL3)

This register is used to determine the vertical and horizontal wait counts.

This register determines the back porches of HSYNC and VSYNC signals when they are generated by the block.

Address: 20F\_8000h base + A0h offset = 20F\_80A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				MUX_SYNC_SIGNALS	VSYNC_ONLY	HORIZONTAL_WAIT_CNT									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	VERTICAL_WAIT_CNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### LCDIF\_VDCTRL3 field descriptions

Field	Description
31–30 RSRVD0	This field is reserved. Reserved bits, write as 0.
29 MUX_SYNC_SIGNALS	When this bit is set, the eLCDIF block will internally mux HSYNC with LCD_D14, DOTCLK with LCD_D13 and ENABLE with LCD_D12, otherwise these signals will go out on separate pins. This feature can be used to maintain backward compatability with 37xx.
28 VSYNC_ONLY	This bit must be set to 1 in the VSYNC mode of operation, and 0 in the DOTCLK mode of operation.
27–16 HORIZONTAL_WAIT_CNT	In the DOTCLK mode, wait for this number of clocks from falling edge (or rising if HSYNC_POL is 1) of HSYNC signal to account for horizontal back porch plus the number of DOTCLKs before the moving picture information begins.
15–0 VERTICAL_WAIT_CNT	In the VSYNC interface mode, wait for this number of CLK_DIS_LCDIFn cycles from the falling VSYNC edge (or rising if VSYNC_POL is 1) before starting LCD transactions and is applicable only if WAIT_FOR_VSYNC_EDGE is set. Minimum is CMD_SETUP+5. In the DOTCLK mode, it accounts for the veritcal back porch lines plus the number of horizontal lines before the moving picture begins. The unit for this parameter is inherently the same as the VSYNC_PERIOD_UNIT.

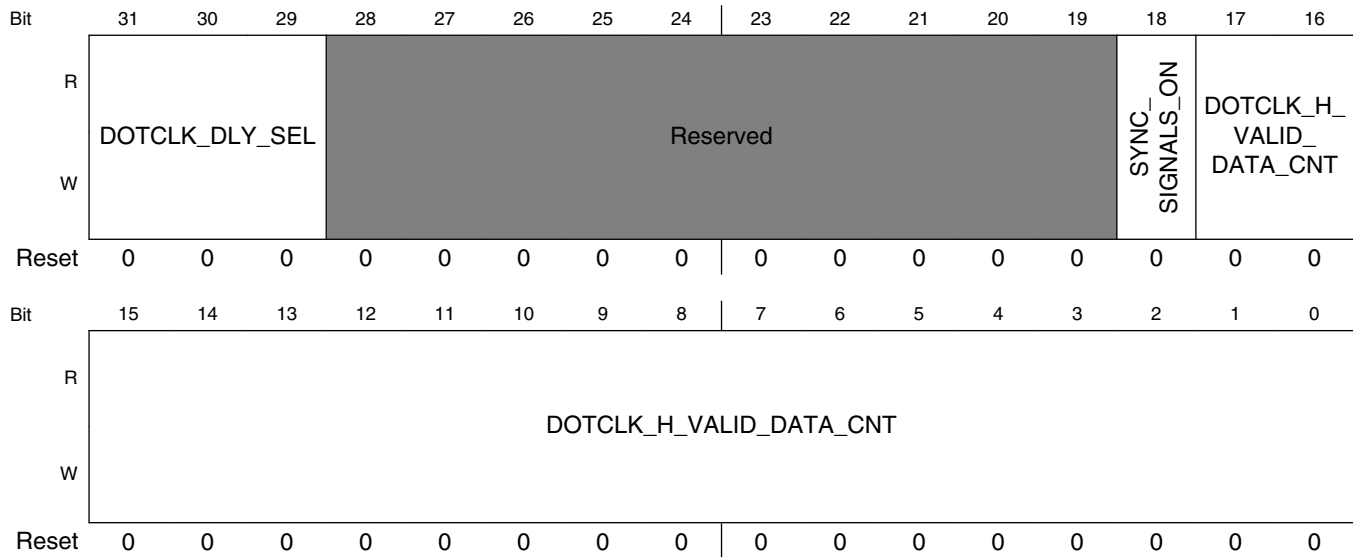
## 21.6.12 eLCDIF VSYNC Mode and Dotclk Mode Control Register4 (LCDIF\_VDCTRL4)

This register is used to control the DOTCLK mode of the block.

This register determines the active data in each horizontal line in the DOTCLK mode.

Note that the total number of active horizontal lines in the DOTCLK mode is the same as the V\_COUNT bitfield in the LCDIF\_TRANSFER\_COUNT register.

Address: 20F\_8000h base + B0h offset = 20F\_80B0h



### LCDIF\_VDCTRL4 field descriptions

Field	Description
31–29 DOTCLK_DLY_SEL	This bitfield selects the amount of time by which the DOTCLK signal should be delayed before coming out of the LCD_DOTCK pin. 0 = 2ns; 1=4ns;2=6ns;3=8ns. Remaining values are reserved.
28–19 RSRVD0	This field is reserved. Reserved bits, write as 0.
18 SYNC_ SIGNALS_ON	Set this field to 1 if the LCD controller requires that the VSYNC or VSYNC/HSYNC/DOTCLK control signals should be active atleast one frame before the data transfers actually start and remain active atleast one frame after the data transfers end. The hardware does not count the number of frames automatically. Rather, the VSYNC edge interrupt can be monitored by software to count the number of frames that have occurred after this bit is set and then the RUN bit can be set to start the data transactions. This bit must always be set in the DOTCLK mode of operation, and it must be set in the VSYNC mode of operation when VSYNC signal is an output.
17–0 DOTCLK_H_ VALID_DATA_ CNT	Total number of CLK_DIS_LCDIFn cycles on each horizontal line that carry valid data in DOTCLK mode.

### 21.6.13 Digital Video Interface Control0 Register (LCDIF\_DVICTRL0)

The Digital Video interface Control0 register provides the overall control of the Digital Video interface.

This register gives information about the horizontal active, horizontal blanking and total number of lines in the ITU-R BT.656 interface.

#### EXAMPLE

```
//525/60 video system
HW_LCDIF_DVICTRL0_H_ACTIVE_CNT_WR(0x5A0); //1440
HW_LCDIF_DVICTRL0_H_BLANKING_CNT_WR(0x106); //262
//625/50 video system
HW_LCDIF_DVICTRL0_H_ACTIVE_CNT_WR(0x5A0); //1440
HW_LCDIF_DVICTRL0_H_BLANKING_CNT_WR(0x112); //274
```

Address: 20F\_8000h base + C0h offset = 20F\_80C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W	Reserved				H_ACTIVE_CNT												Reserved				H_BLANKING_CNT											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### LCDIF\_DVICTRL0 field descriptions

Field	Description
31–28 RSRVD1	This field is reserved. Reserved bits, write as 0.
27–16 H_ACTIVE_CNT	Number of active video samples to be transmitted. (Mostly will be 1440 for both PAL and NTSC). Must always be a multiple of 4.
15–12 RSRVD0	This field is reserved. Reserved bits, write as 0.
11–0 H_BLANKING_CNT	Number of blanking samples to be inserted between EAV and SAV during horizontal blanking interval.

### 21.6.14 Digital Video Interface Control1 Register (LCDIF\_DVICTRL1)

The Digital Video interface Control1 register provides the overall control of the Digital Video interface.

This register contains information about the Field1 start and end, and the Field2 start in the ITU-R BT.656 interface.

**EXAMPLE**

```
//525/60 video system
HW_LCDIF_DVICTRL1_F1_START_LINE_WR(0x4); //4
HW_LCDIF_DVICTRL1_F1_END_LINE_WR(0x109); //265
HW_LCDIF_DVICTRL1_F2_START_LINE_WR(0x10A); //266
//625/50 video system
HW_LCDIF_DVICTRL1_F1_START_LINE_WR(0x1); //1
HW_LCDIF_DVICTRL1_F1_END_LINE_WR(0x138); //312
HW_LCDIF_DVICTRL1_F2_START_LINE_WR(0x139); //313
```

Address: 20F\_8000h base + D0h offset = 20F\_80D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		F1_START_LINE										F1_END_LINE			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	F1_END_LINE						F2_START_LINE									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**LCDIF\_DVICTRL1 field descriptions**

Field	Description
31–30 RSRVD0	This field is reserved. Reserved bits, write as 0.
29–20 F1_START_LINE	Vertical line number from which Field 1 begins.
19–10 F1_END_LINE	Vertical line number at which Field1 ends.
9–0 F2_START_LINE	Vertical line number from which Field 2 begins.

**21.6.15 Digital Video Interface Control2 Register (LCDIF\_DVICTRL2)**

The Digital Video interface Control2 register provides the overall control of the Digital Video interface.

This register contains information about the Field2 end, and the Vertical Blanking1 interval in the ITU-R BT.656 interface.

**EXAMPLE**

```
//525/60 video system
HW_LCDIF_DVICTRL2_F2_END_LINE_WR(0x3); //3
HW_LCDIF_DVICTRL2_V1_BLANK_START_LINE_WR(0x108); //264
HW_LCDIF_DVICTRL2_V1_BLANK_END_LINE_WR(0x11A); //282
//625/50 video system
HW_LCDIF_DVICTRL2_F2_END_LINE_WR(0x271); //625
```

## ELCDIF Memory Map/Register Definition

```
HW_LCDIF_DVICTRL2_V1_BLANK_START_LINE_WR(0x137); //311
HW_LCDIF_DVICTRL2_V1_BLANK_END_LINE_WR(0x14F); //335
```

Address: 20F\_8000h base + E0h offset = 20F\_80E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			F2_END_LINE										V1_BLANK_START_LINE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	V1_BLANK_START_LINE							V1_BLANK_END_LINE								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### LCDIF\_DVICTRL2 field descriptions

Field	Description
31–30 RSRVD0	This field is reserved. Reserved bits, write as 0.
29–20 F2_END_LINE	Vertical line number at which Field 2 ends.
19–10 V1_BLANK_START_LINE	Vertical line number towards the end of Field1 where first Vertical Blanking interval starts.
9–0 V1_BLANK_END_LINE	Vertical line number in the beginning part of Field2 where first Vertical Blanking interval ends.

## 21.6.16 Digital Video Interface Control3 Register (LCDIF\_DVICTRL3)

The Digital Video interface Control3 register provides the overall control of the Digital Video interface.

This register contains information about the Vertical Blanking2 interval in the ITU-R BT. 656 interface.

### EXAMPLE

```
//525/60 video system
HW_LCDIF_DVICTRL3_V2_BLANK_START_LINE_WR(0x1); //1
HW_LCDIF_DVICTRL3_V2_BLANK_END_LINE_WR(0x13); //19
HW_LCDIF_DVICTRL0_V_LINES_CNT_WR(0x20D); //525
//625/50 video system
HW_LCDIF_DVICTRL3_V2_BLANK_START_LINE_WR(0x270); //624
HW_LCDIF_DVICTRL3_V2_BLANK_END_LINE_WR(0x16); //22
HW_LCDIF_DVICTRL0_V_LINES_CNT_WR(0x271); //625
```

Address: 20F\_8000h base + F0h offset = 20F\_80F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	Reserved			V2_BLANK_START_LINE										V2_BLANK_END_LINE			
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	V2_BLANK_END_LINE							V_LINES_CNT								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**LCDIF\_DVICTRL3 field descriptions**

Field	Description
31–30 RSRVD0	This field is reserved. Reserved bits, write as 0.
29–20 V2_BLANK_START_LINE	Vertical line number towards the end of Field2 where second Vertical Blanking interval starts.
19–10 V2_BLANK_END_LINE	Vertical line number in the beginning part of Field1 where second Vertical Blanking interval ends.
9–0 V_LINES_CNT	Total number of vertical lines per frame (generally 525 or 625)

**21.6.17 Digital Video Interface Control4 Register (LCDIF\_DVICTRL4)**

The Digital Video interface Control4 register provides the overall control of the Digital Video interface.

This register is used to add side borders to the output if the input frame width is less than 720 pixels.

**EXAMPLE**

```
//If input frame has only 640 pixels per line, but output is supposed to have
720 pixels per line.
HW_LCDIF_DVICTRL4_H_FILL_CNT_WR(0x50); //80
HW_LCDIF_DVICTRL4_Y_FILL_VALUE_WR(0x10); //16
HW_LCDIF_DVICTRL4_CB_FILL_VALUE_WR(0x80); //128
HW_LCDIF_DVICTRL4_CR_FILL_VALUE_WR(0x80); //128
```

Address: 20F\_8000h base + 100h offset = 20F\_8100h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Y_FILL_VALUE								CB_FILL_VALUE								CR_FILL_VALUE								H_FILL_CNT							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**LCDIF\_DVICTRL4 field descriptions**

Field	Description
31–24 Y_FILL_VALUE	Value of Y component of filler data
23–16 CB_FILL_VALUE	Value of CB component of filler data
15–8 CR_FILL_VALUE	Value of CR component of filler data.
7–0 H_FILL_CNT	Number of active video samples that have to be filled with the filler data in the front and back portions of the active horizontal interval. Must be a multiple of 4. This field will have to be programmed if the input frame has less than 720 pixels per line.

**21.6.18 RGB to YCbCr 4:2:2 CSC Coefficient0 Register (LCDIF\_CSC\_COEFF0)**

LCDIF\_CSC\_COEFF0 register provides overall control over color space conversion from RGB to 4:2:2 YCbCr. The equations for the conversion are given by:  $Y = C0 \cdot R + C1 \cdot G + C2 \cdot B + Y\_offset$   $Cb = C3 \cdot R + C4 \cdot G + C5 \cdot B + CbCr\_offset$   $Cr = C6 \cdot R + C7 \cdot G + C8 \cdot B + CbCr\_offset$

This register carries programming information about RGB to YCbCr 4:2:2 CSC.

**EXAMPLE**

```
HW_LCDIF_CSC_COEFF0_C0_WR(0x41); // 0.257x256=65
HW_LCDIF_CSC_COEFF0_CSC_SUBSAMPLE_FILTER_WR(0x3);
```

Address: 20F\_8000h base + 110h offset = 20F\_8110h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved							C0								
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	Reserved															CSC_ SUBSAMPL E_FILTER	
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**LCDIF\_CSC\_COEFF0 field descriptions**

Field	Description
31–26 RSRVD1	This field is reserved. Reserved bits, write as 0.
25–16 C0	Two's complement red multiplier coefficient for Y

Table continues on the next page...



**LCDIF\_CSC\_COEFF0 field descriptions (continued)**

Field	Description
15–2 RSRVD0	This field is reserved. Reserved bits, write as 0.
1–0 CSC_ SUBSAMPLE_ FILTER	This register describes the filtering and subsampling scheme to be performed on the chroma components in order to convert from YCbCr 4:4:4 to YCbCr 4:2:2 space. Note that the following descriptions apply individually to Cb and Cr.  0x0 <b>SAMPLE_AND_HOLD</b> — No filtering, simply keep every chroma value for samples numbered 2n and discard chroma values associated with all samples numbered 2n+1.  0x1 <b>RSRVD</b> — Reserved  0x2 <b>INTERSTITIAL</b> — Chroma samples numbered 2n and 2n+1 are averaged (weights 1/2, 1/2) and that chroma value replaces the two chroma values at 2n and 2n+1. This chroma now exists horizontally halfway between the two luma samples.  0x3 <b>COSITED</b> — Chroma samples numbered 2n-1, 2n, and 2n+1 are averaged (weights 1/4, 1/2, 1/4) and that chroma value exists at the same site as the luma sample numbered 2n and the chroma samples at 2n+1 are discarded.

**21.6.19 RGB to YCbCr 4:2:2 CSC Coefficient1 Register (LCDIF\_CSC\_COEFF1)**

LCDIF\_CSC\_COEFF1 register provides overall control over color space conversion from RGB to 4:2:2 YCbCr. The equations for the conversion are given by:  $Y = C0 \cdot R + C1 \cdot G + C2 \cdot B + Y\_offset$   $Cb = C3 \cdot R + C4 \cdot G + C5 \cdot B + CbCr\_offset$   $Cr = C6 \cdot R + C7 \cdot G + C8 \cdot B + CbCr\_offset$

This register carries programming information about RGB to YCbCr 4:2:2 CSC.

**EXAMPLE**

```
HW_LCDIF_CSC_COEFF1_C1_WR(0x81) ; // 0.504x256=129
HW_LCDIF_CSC_COEFF1_C2_WR(0x19) ; // 0.098x256=25
```

Address: 20F\_8000h base + 120h offset = 20F\_8120h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						C2										Reserved						C1									
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**LCDIF\_CSC\_COEFF1 field descriptions**

Field	Description
31–26 RSRVD1	This field is reserved. Reserved bits, write as 0.
25–16 C2	Two's complement blue multiplier coefficient for Y

*Table continues on the next page...*

**LCDIF\_CSC\_COEFF1 field descriptions (continued)**

Field	Description
15–10 RSRVD0	This field is reserved. Reserved bits, write as 0.
9–0 C1	Two's complement green multiplier coefficient for Y

**21.6.20 RGB to YCbCr 4:2:2 CSC Coefficient2 Register (LCDIF\_CSC\_COEFF2)**

LCDIF\_CSC\_COEFF2 register provides overall control over color space conversion from RGB to 4:2:2 YCbCr. The equations for the conversion are given by:  $Y = C0 \cdot R + C1 \cdot G + C2 \cdot B + Y\_offset$   $Cb = C3 \cdot R + C4 \cdot G + C5 \cdot B + CbCr\_offset$   $Cr = C6 \cdot R + C7 \cdot G + C8 \cdot B + CbCr\_offset$

This register carries programming information about RGB to YCbCr 4:2:2 CSC.

**EXAMPLE**

```
HW_LCDIF_CSC_COEFF2_C3_WR(0x3DB); // -0.148x256 = -37
HW_LCDIF_CSC_COEFF2_C4_WR(0x3B6); // -0.291x256 = -74
```

Address: 20F\_8000h base + 130h offset = 20F\_8130h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						C4										Reserved						C3									
W	Reserved						C4										Reserved						C3									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**LCDIF\_CSC\_COEFF2 field descriptions**

Field	Description
31–26 RSRVD1	This field is reserved. Reserved bits, write as 0.
25–16 C4	Two's complement green multiplier coefficient for Cb
15–10 RSRVD0	This field is reserved. Reserved bits, write as 0.
9–0 C3	Two's complement red multiplier coefficient for Cb

### 21.6.21 RGB to YCbCr 4:2:2 CSC Coefficient3 Register (LCDIF\_CSC\_COEFF3)

LCDIF\_CSC\_COEFF3 register provides overall control over color space conversion from RGB to 4:2:2 YCbCr. The equations for the conversion are given by:  $Y = C0 \cdot R + C1 \cdot G + C2 \cdot B + Y\_offset$   $Cb = C3 \cdot R + C4 \cdot G + C5 \cdot B + CbCr\_offset$   $Cr = C6 \cdot R + C7 \cdot G + C8 \cdot B + CbCr\_offset$

This register carries programming information about RGB to YCbCr 4:2:2 CSC.

#### EXAMPLE

```
HW_LCDIF_CSC_COEFF3_C5_WR(0x70); //0.439x256=112
HW_LCDIF_CSC_COEFF3_C6_WR(0x70); //0.439x256=112
```

Address: 20F\_8000h base + 140h offset = 20F\_8140h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						C6										Reserved						C5									
W	Reserved						C6										Reserved						C5									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### LCDIF\_CSC\_COEFF3 field descriptions

Field	Description
31–26 RSRVD1	This field is reserved. Reserved bits, write as 0.
25–16 C6	Two's complement red multiplier coefficient for Cr
15–10 RSRVD0	This field is reserved. Reserved bits, write as 0.
9–0 C5	Two's complement blue multiplier coefficient for Cb

### 21.6.22 RGB to YCbCr 4:2:2 CSC Coefficient4 Register (LCDIF\_CSC\_COEFF4)

LCDIF\_CSC\_COEFF4 register provides overall control over color space conversion from RGB to 4:2:2 YCbCr. The equations for the conversion are given by:  $Y = C0 \cdot R + C1 \cdot G + C2 \cdot B + Y\_offset$   $Cb = C3 \cdot R + C4 \cdot G + C5 \cdot B + CbCr\_offset$   $Cr = C6 \cdot R + C7 \cdot G + C8 \cdot B + CbCr\_offset$

This register carries programming information about RGB to YCbCr 4:2:2 CSC.

#### EXAMPLE

```
HW_LCDIF_CSC_COEFF4_C7_WR(0x3A2); //-0.368x256=-94
HW_LCDIF_CSC_COEFF4_C8_WR(0x3EE); //-0.071x256=-18
```

Address: 20F\_8000h base + 150h offset = 20F\_8150h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						C8										Reserved						C7									
W	Reserved						C8										Reserved						C7									
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## LCDIF\_CSC\_COEFF4 field descriptions

Field	Description
31–26 RSRVD1	This field is reserved. Reserved bits, write as 0.
25–16 C8	Two's complement blue multiplier coefficient for Cr
15–10 RSRVD0	This field is reserved. Reserved bits, write as 0.
9–0 C7	Two's complement green multiplier coefficient for Cr

## 21.6.23 RGB to YCbCr 4:2:2 CSC Offset Register (LCDIF\_CSC\_OFFSET)

LCDIF\_CSC\_ register provides overall control over color space conversion from RGB to 4:2:2 YCbCr. The equations for the conversion are given by:  $Y = C0 \cdot R + C1 \cdot G + C2 \cdot B + Y\_offset$   
 $Cb = C3 \cdot R + C4 \cdot G + C5 \cdot B + CbCr\_offset$   
 $Cr = C6 \cdot R + C7 \cdot G + C8 \cdot B + CbCr\_offset$

This register carries programming information about RGB to YCbCr 4:2:2 CSC.

Address: 20F\_8000h base + 160h offset = 20F\_8160h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								CBCR_OFFSET								Reserved								Y_OFFSET							
W	Reserved								CBCR_OFFSET								Reserved								Y_OFFSET							
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

## LCDIF\_CSC\_OFFSET field descriptions

Field	Description
31–25 RSRVD1	This field is reserved. Reserved bits, write as 0.
24–16 CBCR_OFFSET	Two's complement offset for the Cb and Cr components
15–9 RSRVD0	This field is reserved. Reserved bits, write as 0.

Table continues on the next page...

**LCDIF\_CSC\_OFFSET field descriptions (continued)**

Field	Description
8–0 Y_OFFSET	Two's complement offset for the Y component

**21.6.24 RGB to YCbCr 4:2:2 CSC Limit Register (LCDIF\_CSC\_LIMIT)**

LCDIF\_CSC\_CTRL0 register provides overall control over color space conversion from RGB to 4:2:2 YCbCr. The equations for the conversion are given by:  $Y = C0*R + C1*G + C2*B + Y\_offset$   $Cb = C3*R + C4*G + C5*B + CbCr\_offset$   $Cr = C6*R + C7*G + C8*B + CbCr\_offset$

This register carries programming information about RGB to YCbCr 4:2:2 CSC. Note that the values in this register are unsigned.

**EXAMPLE**

```
HW_LCDIF_CSC_LIMIT_CBCR_MIN_WR(0x10); //16
HW_LCDIF_CSC_LIMIT_CBCR_MAX_WR(0xF0); //240
HW_LCDIF_CSC_LIMIT_Y_MIN_WR(0x10); //16
HW_LCDIF_CSC_LIMIT_Y_MAX_WR(0xEB); //235
```

Address: 20F\_8000h base + 170h offset = 20F\_8170h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CBCR_MIN								CBCR_MAX								Y_MIN								Y_MAX							
W																																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1

**LCDIF\_CSC\_LIMIT field descriptions**

Field	Description
31–24 CBCR_MIN	Lower limit of Cb and Cr after RGB to 4:2:2 YCbCr conversion
23–16 CBCR_MAX	Upper limit of Cb and Cr after RGB to 4:2:2 YCbCr conversion
15–8 Y_MIN	Lower limit of Y after RGB to 4:2:2 YCbCr conversion
7–0 Y_MAX	Upper limit of Y after RGB to 4:2:2 YCbCr conversion

## 21.6.25 LCD Interface Data Register (LCDIF\_DATA)

This register is used to transfer data using the PIO interface mode of operation. In MPU mode, data written to this register will be transferred out to the display device. When receiving data from the display, data is read from this register using PIO operations. During write operations, data can be written to this register (from the processor's perspective) as bytes, half-words (16 bits), or words (32 bits) as desired.

This register holds the 32-bit word written by the ARM platform into LCDIF. This data then gets sent out by the block across the interface.

Address: 20F\_8000h base + 180h offset = 20F\_8180h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### LCDIF\_DATA field descriptions

Field	Description
31–24 DATA_THREE	Byte 3 (most significant byte) of data written to LCDIF.
23–16 DATA_TWO	Byte 2 of data written to eLCDIF.
15–8 DATA_ONE	Byte 1 of data written to eLCDIF.
7–0 DATA_ZERO	Byte 0 (least significant byte) of data written to eLCDIF.

## 21.6.26 Bus Master Error Status Register (LCDIF\_BM\_ERROR\_STAT)

This register reflects the virtual address at which the AXI master received an error response from the slave.

When the BM\_ERROR\_IRQ is asserted, the address of the bus error is updated in the register.

Address: 20F\_8000h base + 190h offset = 20F\_8190h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**LCDIF\_BM\_ERROR\_STAT field descriptions**

Field	Description
31–0 ADDR	Virtual address at which bus master error occurred.

**21.6.27 CRC Status Register (LCDIF\_CRC\_STAT)**

This register reflects the CRC value of each frame sent out by eLCDIF. The CRC is done on the final output bus, so the value will be dependent on the LCD\_DATABUS\_WIDTH bitfield even if the input data is the same.

This register will be updated when the CUR\_FRAME\_DONE\_IRQ is asserted. In the case of DVI mode, the CRC is calculated for the entire frame, not separately for each field in the frame.

Address: 20F\_8000h base + 1A0h offset = 20F\_81A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	CRC_VALUE																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**LCDIF\_CRC\_STAT field descriptions**

Field	Description
31–0 CRC_VALUE	Calculated CRC value.

**21.6.28 LCD Interface Status Register (LCDIF\_STAT)**

The LCD interface status register can be used to check the current status of the eLCDIF block.

The LCD interface status register that contains read only views of some parameters or current state of the block.

ELCDIF Memory Map/Register Definition

Address: 20F\_8000h base + 1B0h offset = 20F\_81B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PRESENT	Reserved	LFIFO_FULL	LFIFO_EMPTY	TXFIFO_FULL	TXFIFO_EMPTY	BUSY	DVI_CURRENT_FIELD	Reserved							
W																
Reset	1	0	0	1	0	1	0	1	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								LFIFO_COUNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



**LCDIF\_STAT field descriptions**

Field	Description
31 PRESENT	0: eLCDIF not present on this product 1: eLCDIF is present.
30 -	This field is reserved. Reserved.
29 LFIFO_FULL	Read only view of the signal that indicates that LCD read datapath FIFO is full, will be generally used in the write mode of the LCD interface.
28 LFIFO_EMPTY	Read only view of the signal that indicates that LCD read datapath FIFO is empty, will be generally used in the read mode of the LCD interface.
27 TXFIFO_FULL	Read only view of the signal that indicates that LCD write datapath FIFO is full, will be generally used in the write mode of the LCD interface.
26 TXFIFO_EMPTY	Read only view of the signal that indicates that LCD write datapath FIFO is empty, will be generally used in the read mode of the LCD interface.
25 BUSY	Read only view of the input busy signal from the external LCD controller.
24 DVI_CURRENT_FIELD	Read only view of the current field being transmitted. DVI_CURRENT_FIELD = 0 means field 1. DVI_CURRENT_FIELD = 1 means field 2.
23–9 RSRVD0	This field is reserved. Reserved bits. Write as 0.
8–0 LFIFO_COUNT	Read only view of the current count in Latency buffer (LFIFO).

**21.6.29 LCD Interface Version Register (LCDIF\_VERSION)**

The LCD interface version register can be used to read the version of the eLCDIF IP being used in this SoC.

The LCD interface debug register is for diagnostic use only.

Address: 20F\_8000h base + 1C0h offset = 20F\_81C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJOR								MINOR								STEP															
W																																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**LCDIF\_VERSION field descriptions**

Field	Description
31–24 MAJOR	Fixed read-only value reflecting the MAJOR field of RTL version.
23–16 MINOR	Fixed read-only value reflecting the MINOR field of RTL version.
15–0 STEP	Fixed read-only value reflecting the stepping of RTL version.

LCDIF\_VERSION field descriptions (continued)

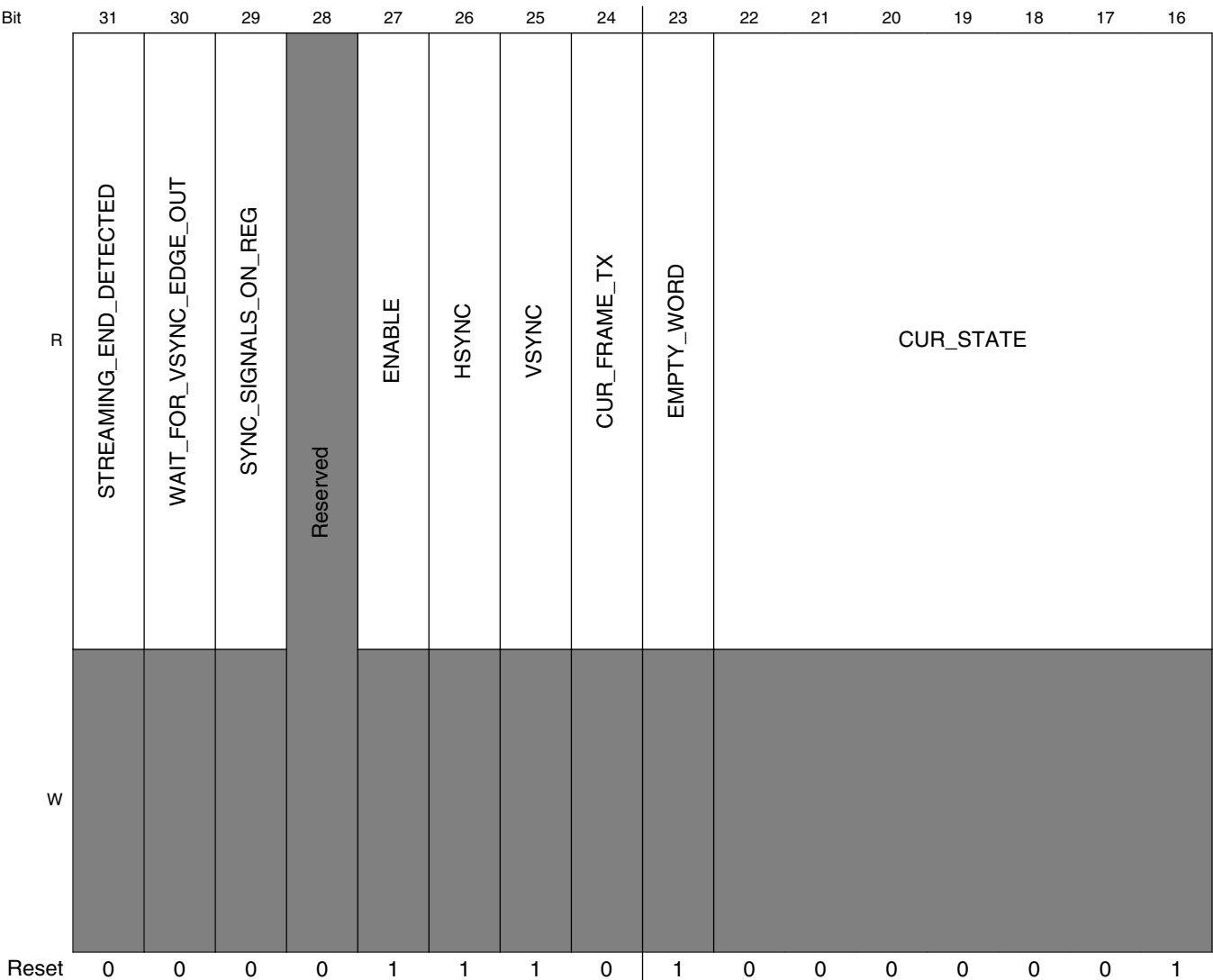
Field	Description
-------	-------------

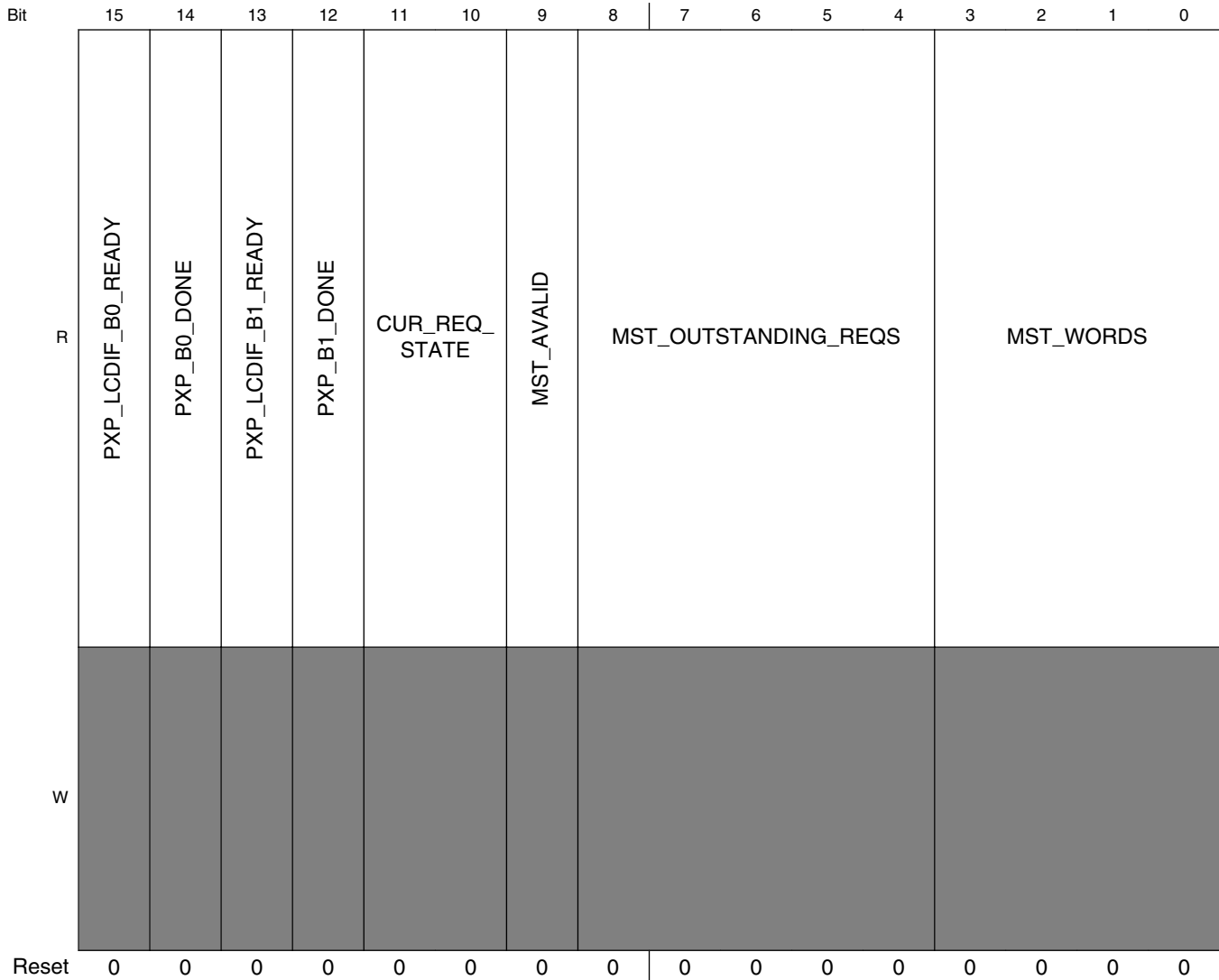
21.6.30 LCD Interface Debug0 Register (LCDIF\_DEBUG0)

The LCD interface debug0 register provides a diagnostic view of the state machine and other useful internal signals.

The LCD interface debug register is for diagnostic use only.

Address: 20F\_8000h base + 1D0h offset = 20F\_81D0h





LCDIF\_DEBUG0 field descriptions

Field	Description
31 STREAMING_ END_DETECTED	Read only view of the DOTCLK_MODE or DVI_MODE bit going from 1 to 0.
30 WAIT_FOR_ VSYNC_EDGE_ OUT	Read only view of WAIT_FOR_VSYNC_EDGE bit in the VSYNC mode after it comes out of the TXFIFO.
29 SYNC_SIGNALS_ ON_REG	Read only view of internal sync_signals_on_reg signal.
28 -	This field is reserved. Reserved.
27 ENABLE	Read only view of ENABLE signal.

Table continues on the next page...

**LCDIF\_DEBUG0 field descriptions (continued)**

Field	Description
26 HSYNC	Read only view of HSYNC signal.
25 VSYNC	Read only view of VSYNC signal.
24 CUR_FRAME_TX	This bit is 1 for the time the current frame is being transmitted in the VSYNC mode. Useful for VSYNC mode debug.
23 EMPTY_WORD	Indicates that the current word is empty.
22–16 CUR_STATE	Read only view of the current state machine state in the current mode of operation.
15 PXP_LCDIF_B0_READY	Buffer0 ready signal issued by ePXP.
14 PXP_B0_DONE	Buffer0 done signal issued by eLCDIF.
13 PXP_LCDIF_B1_READY	Buffer1 ready signal issued by ePXP.
12 PXP_B1_DONE	Buffer1 done signal issued by eLCDIF.
11–10 CUR_REQ_STATE	Read only view of the request state machine.
9 MST_AVALID	Read only view of the mst_avalid signal issued by the AXI bus master.
8–4 MST_OUTSTANDING_REQS	Read only view of the current outstanding requests issued by the AXI bus master.
3–0 MST_WORDS	Read only view of the current bursts issued by the AXI bus master.

**21.6.31 LCD Interface Debug1 Register (LCDIF\_DEBUG1)**

The LCD interface debug1 register provides a diagnostic view of the state machine and other useful internal signals.

The LCD interface debug register is for diagnostic use only.

Address: 20F\_8000h base + 1E0h offset = 20F\_81E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	H_DATA_COUNT																V_DATA_COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**LCDIF\_DEBUG1 field descriptions**

Field	Description
31–16 H_DATA_COUNT	Read only view of the current state of the horizontal data counter.
15–0 V_DATA_COUNT	Read only view of the current state of the vertical data counter.

**21.6.32 LCD Interface Debug2 Register (LCDIF\_DEBUG2)**

The LCD interface debug2 register provides a diagnostic view of the state machine and other useful internal signals.

The LCD interface debug register is for diagnostic use only.

Address: 20F\_8000h base + 1F0h offset = 20F\_81F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MST_ADDRESS																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**LCDIF\_DEBUG2 field descriptions**

Field	Description
31–0 MST_ADDRESS	Read only view of the current address issued by the AXI bus master.

**21.6.33 eLCDIF Threshold Register (LCDIF\_THRES)**

This register is used to activate control signals when the number of pixels reaches the programmed threshold. These control signals, in turn, can be used to manipulate access priority or dynamically change the input clock frequency to meet the required pixel throughput.

Memory request priority threshold register.

Address: 20F\_8000h base + 200h offset = 20F\_8200h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved							FASTCLOCK								Reserved							PANIC									
W																																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

**LCDIF\_THRES field descriptions**

<b>Field</b>	<b>Description</b>
31–25 RSRVD2	This field is reserved. Reserved bits. Write as 0.
24–16 FASTCLOCK	This value should be set to a value of pixels, from 0 to 511. When the number of pixels in the input pixel FIFO is LESS than this value, the fast clock control output will be raised. This signal can be used to reduce the system bus clock frequency to save power during horizontal/vertical blanking intervals. This value should also be programmed to a value that is greater than the "PANIC" threshold value. This will allow a faster clock to recover the number of pixels in the FIFO before a "panic" level is encountered.
15–9 RSRVD1	This field is reserved. Reserved bits. Write as 0.
8–0 PANIC	This value should be set to a value of pixels, from 0 to 511. When the number of pixels in the input pixel FIFO is LESS than this value, the panic control output will be raised. This signal can be used to raise the access eLCDIF's access priority.

# Chapter 22

## Electrophoretic Display Controller (EPDC)

### 22.1 Overview

This chapter describes the architecture of the EPDC. It provides a detailed description of the block for digital design and software driver development.

The EPDC is a feature-rich, low power and high performance direct drive active matrix EPD controller. It is specifically designed to drive E•INK™ EPD panels supporting a wide variety of TFT backplanes. The goal of the EPDC is to provide an efficient SoC integration of this functionality for e-paper applications, allowing a significant BOM cost saving over an external solution, while reaching much higher levels of performance at lower power. The EPDC module is defined in the context of an optimized HW/SW partitioning and works in conjunction with the PXP IP module to form a complete display processing solution.

The key features of the EPDC are as follows:

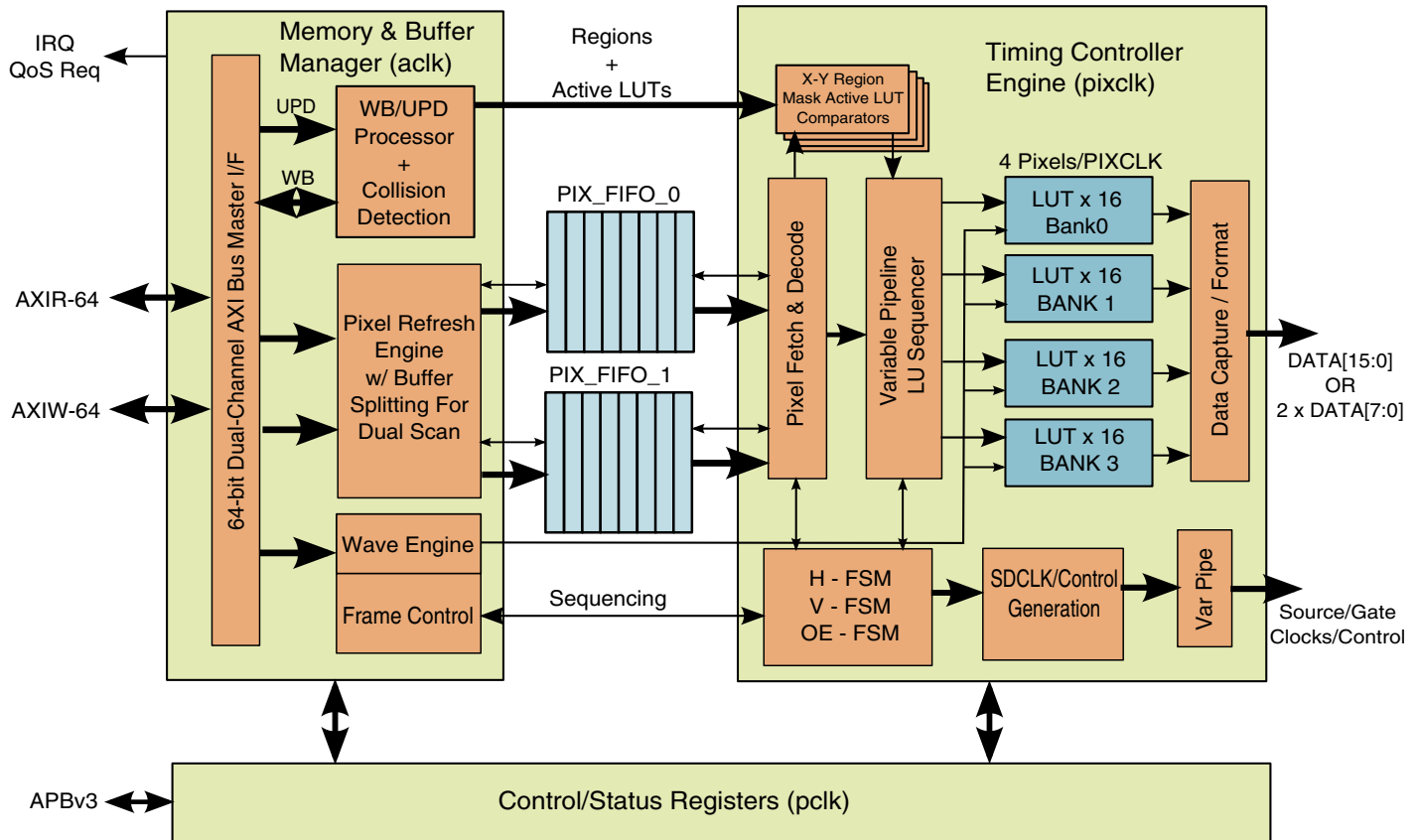
- TFT resolutions up to 4096 x 4096 pixels with 20 Hz refresh (programmable up to 8191 x 8191)
- TFT resolutions up to 1650 x 2332 pixels at 106 Hz refresh
- Industry standard bus interfaces (AMBA AXI and APB)
- Up to 5-bit pixel representation for up to 32 greyscale levels
- Up to 64 concurrent updates with partial update support, except for 32(5-bit) grey level panel for which only 16 concurrent updates can be used
- Automatic collision handling when used in conjunction with the i.MX device driver
- Dual-scan TFT drive mode to support ultra high resolution/refresh rate displays
- Flexible direct drive TFT interface supporting next generation source driver, gate driver and panel architectures, including LVDS, DDR and multi-level source drivers
- Unified generic configurable timing mode (Pigeon Mode) available on most panel timing control signals
- High performance pixel pipeline architecture to guarantee refresh performance at high pixel rates without the need for high internal clocking

- Ability to process multiple updates asynchronously to refresh/update operations with ability to intercept each frame scan with multiple update requests
- Performance tuning capabilities which can interface with SoC level memory arbitration mechanisms further guaranteeing frame refresh operation
- Decoupled clocking architecture allowing for independent and asynchronous clock sources for memory bus, peripheral bus and pixel clock domains
- Full and partial update mode support
- Support for up to 256 waveform modes, also support optimal waveform auto-selection based on grey level of the pixels being updated
- Low power mode operation via architectural clock gating
- Update buffer analysis functions to get information like collision rectangle, grey level

### 22.1.1 EPDC Block Diagram

The top-level view of the EPDC is shown in the following figure.





**Figure 22-1. EPDC Top-Level Block Diagram**

The EPD function can be separated into two major asynchronous processes. The first is the front end portion which is responsible for processing display updates. The second is the function of driving waveforms to the TFT panel in order to update the screen contents (the refresh operation). The EPDC is comprised of two major submodules (MBM and TCE) which mirror the architectural split in the EPD algorithm. These submodules communicate through a number of control/sequencing signals and data (through the pixel FIFOs and LUT memories):

- **Memory and Buffer Manager (MBM):** This submodule is responsible for all memory-related operations and LUT life cycle control, which involves the frame count management for all updates. This module is clocked by both *aclk* and *pixclk*. It also encapsulates the APB register interface which is clocked by *aclk* too.
- **Timing Control Engine (TCE):** This submodule is responsible for performing TFT scan frame refreshes. It is clocked by *pixclk*. It also houses the LUT memory system.

## 22.2 External Signals

The following table describes the external signals of EPDC:

**Table 22-1. EPDC External Signals**

Signal	Description	Pad	Mode	Direction
EPDC_BDR0	Panel-Border Control (SW controlled)	ECSPI1_MISO	ALT3	O
		EPDC_BDR0	ALT0	
EPDC_BDR1	Panel-Border Control (SW controlled)	ECSPI1_SS0	ALT3	O
		EPDC_BDR1	ALT0	
EPDC_DATA00	Source Driver-Shift signal	EPDC_D0	ALT0	O
EPDC_DATA01	Source Driver-Shift signal	EPDC_D1	ALT0	O
EPDC_DATA02	Source Driver-Shift signal	EPDC_D2	ALT0	O
EPDC_DATA03	Source Driver-Shift signal	EPDC_D3	ALT0	O
EPDC_DATA04	Source Driver-Shift signal	EPDC_D4	ALT0	O
EPDC_DATA05	Source Driver-Shift signal	EPDC_D5	ALT0	O
EPDC_DATA06	Source Driver-Shift signal	EPDC_D6	ALT0	O
EPDC_DATA07	Source Driver-Shift signal	EPDC_D7	ALT0	O
EPDC_DATA08	Source Driver-Shift signal	EPDC_D8	ALT0	O
EPDC_DATA09	Source Driver-Shift signal	EPDC_D9	ALT0	O
EPDC_DATA10	Source Driver-Shift signal	EPDC_D10	ALT0	O
EPDC_DATA11	Source Driver-Shift signal	EPDC_D11	ALT0	O
EPDC_DATA12	Source Driver-Shift signal	EPDC_D12	ALT0	O
EPDC_DATA13	Source Driver-Shift signal	EPDC_D13	ALT0	O
EPDC_DATA14	Source Driver-Shift signal	EPDC_D14	ALT0	O
EPDC_DATA15	Source Driver-Shift signal	EPDC_D15	ALT0	O
EPDC_GDCLK	Gate Driver-Clock	EPDC_GDCLK	ALT0	O
EPDC_GDOE	Gate Driver-Output Enable	EPDC_GDOE	ALT0	O
EPDC_GDRL	Gate Driver-Shift direction	EPDC_GDRL	ALT0	O
EPDC_GDSP	Gate Driver-Start Pulse	EPDC_GDSP	ALT0	O
EPDC_PWR_COM	Panel-Power control (SW controlled)	EPDC_D12	ALT2	O
		EPDC_PWRCOM	ALT0	
EPDC_PWR_CTRL0	Panel-Power control (SW controlled)	EPDC_D8	ALT2	O
		EPDC_PWRCTRL0	ALT0	
EPDC_PWR_CTRL1	Panel-Power control (SW controlled)	EPDC_D9	ALT2	O
		EPDC_PWRCTRL1	ALT0	
EPDC_PWR_CTRL2	Panel-Power control (SW controlled)	EPDC_D10	ALT2	O
		EPDC_PWRCTRL2	ALT0	
EPDC_PWR_CTRL3	Panel-Power control (SW controlled)	EPDC_D11	ALT2	O
		EPDC_PWRCTRL3	ALT0	
EPDC_PWR_IRQ	Panel-Power irq	EPDC_D13	ALT2	I
		EPDC_PWRINT	ALT0	

Table continues on the next page...

**Table 22-1. EPDC External Signals (continued)**

Signal	Description	Pad	Mode	Direction
EPDC_PWR_STAT	Panel-Power status good	EPDC_D14	ALT2	I
		EPDC_PWRSTAT	ALT0	
EPDC_PWR_WAKE	Panel-Power control wake signal (SW controlled)	EPDC_D15	ALT2	O
		EPDC_PWRWAKEUP	ALT0	
EPDC_SDCE0	Source Driver-Chip-enable/Start-Pulse	EPDC_SDCE0	ALT0	O
EPDC_SDCE1	Source Driver-Chip-enable/Start-Pulse	EPDC_SDCE1	ALT0	O
EPDC_SDCE2	Source Driver-Chip-enable/Start-Pulse	EPDC_SDCE2	ALT0	O
EPDC_SDCE3	Source Driver-Chip-enable/Start-Pulse	EPDC_SDCE3	ALT0	O
EPDC_SDCE4	Source Driver-Chip-enable/Start-Pulse	EPDC_SDSHR	ALT2	O
		SD1_CLK	ALT3	
EPDC_SDCE5	Source Driver-Chip-enable/Start-Pulse	EPDC_VCOM0	ALT6	O
		SD1_CMD	ALT3	
EPDC_SDCE6	Source Driver-Chip-enable/Start-Pulse	EPDC_VCOM1	ALT6	O
		SD1_DAT0	ALT3	
EPDC_SDCE7	Source Driver-Chip-enable/Start-Pulse	EPDC_BDR0	ALT6	O
		SD1_DAT1	ALT3	
EPDC_SDCE8	Source Driver-Chip-enable/Start-Pulse	EPDC_BDR1	ALT6	O
		SD1_DAT2	ALT3	
EPDC_SDCE9	Source Driver-Chip-enable/Start-Pulse	SD1_DAT3	ALT3	O
EPDC_SDCLK_N	Negative Source Driver-Shift Clock	SD1_DAT4	ALT3	O
EPDC_SDCLK_P	Positive Source Driver-Shift Clock	EPDC_SDCLK	ALT0	O
EPDC_SDLE	Source Driver-Latch Enable	EPDC_SDLE	ALT0	O
EPDC_SDOE	Source Driver-Output Enable	EPDC_SDOE	ALT0	O
EPDC_SDOED	Source Driver-Output Enable (to VDD)	SD1_DAT5	ALT3	O
EPDC_SDOEZ	Source Driver-Output Enable (to Zero)	SD1_DAT6	ALT3	O
EPDC_SDSHR	Source Driver-Shift dir	EPDC_SDSHR	ALT0	O
EPDC_VCOM0	Panel-VCOM	ECSP11_SCLK	ALT3	O
		EPDC_VCOM0	ALT0	
EPDC_VCOM1	Panel-VCOM	ECSP11_MOSI	ALT3	O
		EPDC_VCOM1	ALT0	

## 22.3 Programming Model

From an application perspective, the EPDC HW/SW solution aims to abstract much of the details of driving an EPD display from the user space.

This is achieved through a blend of HW features and the kernel-level SW driver implementation of the EPD frame buffer. The scope of this chapter mostly pertains to EPDC HW but in order to correctly define the context of the programming model, certain assumptions about the driver architecture (including use of the eXPX IP) are described below.

### 22.3.1 Assumptions

The scope of the EPDC does not include certain functions performed by the driver and application layer.

These features are as follows:

- Maintenance of update and collision lists. This includes, but is not limited to, responsibility for managing update order. Updates are processed in the order they are received by the EPDC. Actual start times of multiple updates can occur on the same frame scan. All pixels contained within an update will always begin on the same frame scan.
- Border control. This function is assumed to be performed by a GPIO functionality.
- EPD Panel Power Management. It is assumed that the driver and application layers will control the panel PMIC functions including relevant interfaces such as I<sup>2</sup>C and 4-wire power sequencing signals.
- Reading and decompression of the waveform file. This is assumed to be unpacked, reformatted and written to system memory into a form that is easily accessed by the EPDC.
- Waveform mode selection. The EPDC provides a mechanism through its histogram analysis feature (which can be run as part of any frame/region processing operations) to allow the SW driver to characterize the region/buffer in terms of gray-map content. For example, EPDC can determine if the update region (considering only those pixels undergoing a transition) contains only 2, 4, 8 or 16 gray levels, HW can automatically select waveform mode according to a programmed LUT holding mapping between grey level and waveform mode. SW driver can also manually choose the appropriate waveform update mode (e.g., when doing an update to repair ghosting artifacts).

## 22.3.2 Register Space (Write/Set/Clear/Toggle)

The EPDC registers utilize four word address locations per 32-bit register.

For control registers (especially interrupts), this provides for a unique address for the following operations:

- Write (0x0). The base address of the register allows full writes to the 32-bit register.
- Set (0x4). This address allows a set operation on the register or its fields; writing a 1 to a register bit/field with the set address will set that bit. Writing a 0 to the set address has no effect.
- Clear (0x8). This address allows a clear operation on the register or its fields; writing a 1 to a register bit/field with the clear address will clear that bit or field.

### NOTE

Interrupt status bits (such as those contained in EPDC\_IRQ) must be cleared with the clear address only. Writing to the EPDC\_IRQ register with a 0 will not clear the interrupt status/source. Writing a 0 to the clear address has no effect.

- Toggle (0xc). This address allows a toggle operation on the register or its fields; writing a 1 to a register bit/field with the toggle address will invert the state of that bit/field. This address is especially useful because it provides the ability to perform a read-modify-write operation with a single write operation. Writing a 0 to the toggle address has no effect.

## 22.3.3 Interrupts

### 22.3.3.1 Interrupt Sources

The EPDC contains 72 ( 64 of LUT completes IRQs and 8 of common IRQs ) unique interrupt sources. Each interrupt source has an interrupt status bit captured in EPDC\_IRQ /EPDC\_IRQ1/EPDC\_IRQ2 .

When an interrupt event occurs, the appropriate bit within the EPDC\_IRQ\* register is set by the EPDC. Each of the bits in EPDC\_IRQ is first AND'ed with its enable bit (in EPDC\_IRQ\_MASK\*), and then the masked bits are logically OR'ed together to generate the final output.

### 22.3.3.2 Enabling/Masking Interrupts

Each of the EPDC interrupt sources can be individually enabled through the bits in the EPDC\_IRQ\_MASK\* register.

Even if an interrupt source is masked out (IRQ\_EN bit set to 0), its status will still be available in the EPDC\_IRQ\* register. The difference is that it does not result in the interrupt line being asserted.

### 22.3.3.3 Handling/Clearing Interrupts

When the software driver is entered as a result of an EPDC interrupt it should read the EPDC\_IRQ\* interrupt status register.

Because multiple interrupts may have fired, in the case of a collision for example, both WB\_CMPLT\_IRQ and LUT\_COL\_IRQ fields of EPDC\_IRQ will be set. Once the interrupt has been handled, the driver should clear the interrupt source by writing to the CLEAR address (see [Register Space \(Write/Set/Clear/Toggle\)](#) for details).

When collision has occurred and the LUT\_COL\_IRQ bit is set, SW will handle the collision and clear LUT\_COL\_IRQ. Please note HW will also clear the entire EPDC\_STATUS\_COL\* register at the sametime on LUT\_COL\_IRQ clearing by SW.

## 22.3.4 Controller Setup

Before screen update operations can be performed, the EPDC should be configured appropriately.

These rudimentary setup operations include:

- Waveform data must be present in memory (in a format that is consistent with the method in which the EPDC accesses it). Because the format of the waveform data is supplier proprietary information, it is not discussed in this chapter.
- Configuring the panel architecture parameters (source and gate-driver configuration)
- Configuring line and frame timing parameters
- Initializing the working buffer and panel

### 22.3.4.1 Memory Requirements

EPDC requires continuous access to the unpacked waveform data, the update region buffer, and its working buffer.

These are all expected to reside in system memory (typically external DRAM). As such, each has particular requirements:

- The Working Buffer (WB), which is pointed to by EPDC\_WB\_ADDR, must have EPDC\_RES[HORIZONTAL] x EPDC\_RES[VERTICAL] x 2 bytes allocated. This buffer is used by the EPDC to process updates and perform TFT refresh operations.
  - EPDC\_RES[HORIZONTAL] mod 4 must equal to 0 for memory allocation calculations because the EPDC constructs each line of the working buffer at a 64-bit boundary.
- The Update Buffer, which is pointed to by EPDC\_UPD\_ADDR (and can be redefined with a different address for each update), must have the following allocation in memory:
  - a. when stride feature not enabled
    - EPDC\_UPD\_SIZE[HEIGHT] x EPDC\_UPD\_SIZE[WIDTH] bytes
    - Note that EPDC\_UPD\_SIZE[WIDTH] mod 8 must equal to 0 for memory allocation purposes. This means that for the address size allocation, the WIDTH field must be rounded up to the nearest number so that it can be divided by 8. This is because the EPDC reads each line of the update buffer at a 64-bit boundary (it should be noted that this is consistent with the ePXP IP output). The actual value programmed for the update can be at the pixel granularity.
  - b. when stride feature enabled
    - EPDC\_UPD\_SIZE[HEIGHT] x EPDC\_UPD\_STRIDE bytes
    - EPDC\_UPD\_SIZE[WIDTH] is used to reflect valid bytes in a line with EPDC\_UPD\_STRIDE pixels, STRIDE >= WIDTH
    - no alignment requirement on line start or end, no padding necessary unless STRIDE > WIDTH
- The waveform data set size is a function of mode count, number of temperature tables and number of frames required for each temperature-compensated mode.

In summary, the memory requirements are as follows (in bytes):

- WB-roundup4(EPDC\_RES[HORIZONTAL]) x EPDC\_RES[VERTICAL] x 2
- when stride feature not enabled: UPD-N x roundup8(EPDC\_UPD\_SIZE[WIDTH]) x EPDC\_UPD\_SIZE[HEIGHT]
  - N = the number of update requests currently being managed by the i.MX EPD driver (that is, the driver may maintain a list of updates in memory which it uses to feed the EPDC).
- when stride feature enabled: UPD-N x EPDC\_UPD\_STRIDE x EPDC\_UPD\_SIZE[HEIGHT]

## 22.3.4.2 Panel Architecture Configuration

The EPDC is designed to directly drive EPD panels which typically expose the source and gate driver IC interfaces (these are often abstracted in traditional displays such as TFT-LCD).

The source and gate driver ICs are responsible for driving the TFT matrix. There are variations in both the panel architectures and the underlying source and gate driver IC functionality.

All the key configuration parameters are defined in the EPDC\_RES, EPDC\_FORMAT, EPDC\_TCE\_CTRL, EPDC\_TCE\_SDCFG and EPDC\_TCE\_GDCFG registers as follows.

- **EPDC\_RES:**
  - **HORIZONTAL:** The panel horizontal resolution (in pixels). The horizontal resolution must be an integer multiple of the source driver PIXELS\_PER\_SDCLK value. It also must be an integer multiple of the PIXELS\_PER\_CE value.
  - **VERTICAL:** The panel vertical resolution (in pixels). This can be any integer value.
- **EPDC\_FORMAT:**
  - **DEFAULT\_TFT\_PIXEL:** This field should almost always be left at 0x00. This register field is used as the value for the TFT pixel during a frame scan when a pixel is not being updated. This condition is commonplace especially when using partial updates or updates which do not encompass the entire screen resolution. This value must always correspond to the state the source driver should see when no voltage change is needed. Incorrectly setting this value will damage the EPD material.
  - **TFT\_PIXEL\_FORMAT:** This enumerated field defines the width of the TFT pixel. The TFT pixel is defined as the per-pixel voltage control value. The most common pixel format is 2B (2 bits per pixel). Panels which support source driver voltage modulation (multi-level voltage control) should use the 4B format. For source drivers that support 7 levels, the 4B format can be used and the unused DATA output pins should be left floating at the board level connection (specifically DATA3, 7, 11, 15). The 2BV and 4BV are variations which require the waveform data to supply a 2-bit VCOM value for each pixel.
- **EPDC\_TCE\_CTRL:**
  - **SDDO\_WIDTH:** This selects 8- or 16-bit DATA operation using enumerations 8-bit and 16-bit. This field is used to describe useful data for panels that support LVDS. DATA[15:8] would be used to drive differential data. In these modes, SDDO\_WIDTH would be set to 8-bit.



- **VCOM\_MODE, VCOM\_VAL:** This VCOM\_MODE bit allows the EPDC to either use the VCOM value supplied in the waveform or a software programmable value defined in the VCOM\_VAL field, for panels which support VCOM modulation.
- **DDR\_MODE:** This mode-bit should be set for panels which expect source-driver data to be available on both edges of the SDCLK. This is common place for panels that support differential signaling, but the feature is not limited to LVDS panels. For example, the EPDC supports DDR modes which make full use of DATA[15:0]. In these modes, DDR\_MODE = 1 and LVDS\_MODE = 0.
- **LVDS\_MODE:** This mode always requires DDR\_MODE to be set. When this mode bit is set, differential signaling is enabled such that DATA[15:8] always drives the inverse of the pixel data which is presented on DATA[7:0]. Because LVDS source-drivers use 2 pins per data-bit, they are typically configured to work in DDR mode. Differential signaling is also supported on the SDCE pins (as an option selected by LVDS\_MODE\_CE). In this mode SDCE[9:5] are used to drive the inverse differential signals for SDCE[4:0].
- **LVDS\_MODE\_CE:** When LVDS\_MODE is set, this bit can also be set to allow SDCE[9:5] to act as differential pairs for each SDCE[4:0].
- **DUAL\_SCAN:** Setting this field enables the dual scan function. When this field is set, all other parameters (except resolution) should be programmed in reference to each half of the panel. Because DUAL\_SCAN modes require 2x the pixel-generation capability, there are limitations to the available TFT modes supported in this configuration (see [Table 22-2](#) for details).
- **SCAN\_DIR0, SCAN\_DIR1:** These fields determine the vertical scan direction of the panel.
  - **SCAN\_DIR0:** Determines the vertical scan direction of the panel when DUAL\_SCAN = 0. When DUAL\_SCAN = 1, it determines the vertical scan direction of the top-half of the panel (0 means scan down and 1 means scan up).
  - **SCAN\_DIR1** only has meaning when DUAL\_SCAN = 1. It defines the vertical scan direction of the bottom half of the panel.
- **PIXELS\_PER\_SDCLK:** This is a key configuration and timing parameter. Each source driver latches a number of TFT pixels per shift clock period (SDCLK). This register defines how many pixels are driven per SDCLK period. When DDR\_MODE = 1, pixels are driven on both edges of the clock, so this value must be adjusted accordingly. See [Table 22-2](#) for the required values per mode.
- **EPDC\_TCE\_SDCFG:**
  - **SDCLK\_HOLD:** Holds the SDCLK low during the LINE\_BEGIN time. The purpose of this field is to allow the user to set a LINE\_BEGIN time which is not an integer multiple of SDCLKs (for fine tuning the line-time).

- SDSHR: Defines the value of the SDSHR output signal which determines the source driver output order mapping.
- NUM\_CE: Defines the total number of source driver chip-enables (must be 1-10). This number doesn't always correspond to the number of source drivers in the panel. Many panels require only a single chip-enable signal which is active during the entire line data time. This signal is often called SPH (Horizontal Start Pulse) in such cases.
- PIXELS\_PER\_CE: This register defines the span of each chip-enable expressed in pixels. For panels that utilize a single global CE, this value should be set to the horizontal resolution of the panel.
- SDDO\_REFORMAT: This enumerated field allows for TFT-pixel level reordering within DATA. For most panels it is recommended to set SDDO\_REFORMAT to the REVERSE\_PIXELS enumeration.
- SDDO\_INVERT: When this field is set, the values on DATA are inverted relative to the values extracted from the waveform LUT operations.
- EPDC\_TCE\_GDCFG:
  - GDRL: Defines the value of the GDRL output signal which determines the gate driver pulse shift direction.
  - GDOE\_MODE: This field selects between two possible methods for driving the GDOE signal. When GDOE\_MODE = 0, the GDOE output is always driven during the frame scan time, except during FRAME\_SYNC when it is driven low. When GDOE\_MODE = 1, the GDOE signal mimics the GDCLK signal but is only active during the frame-data time (FD). For most panels the recommended setting is 0.
  - GDSP\_MODE: This field selects between two possible methods for driving the GDSP signal. When GDSP\_MODE = 0, the GDSP signal is driven on the first line clock time of the FRAME\_BEGIN time. The signal is driven for one GDCLK period (same as line time), and can be shifted using the GDSP\_OFFSET control. When GDSP\_MODE = 1, GDSP is active during the entire FRAME\_SYNC time, and GDSP\_OFFSET has not effect in this mode.

### 22.3.4.3 TFT Panel Timing Configuration

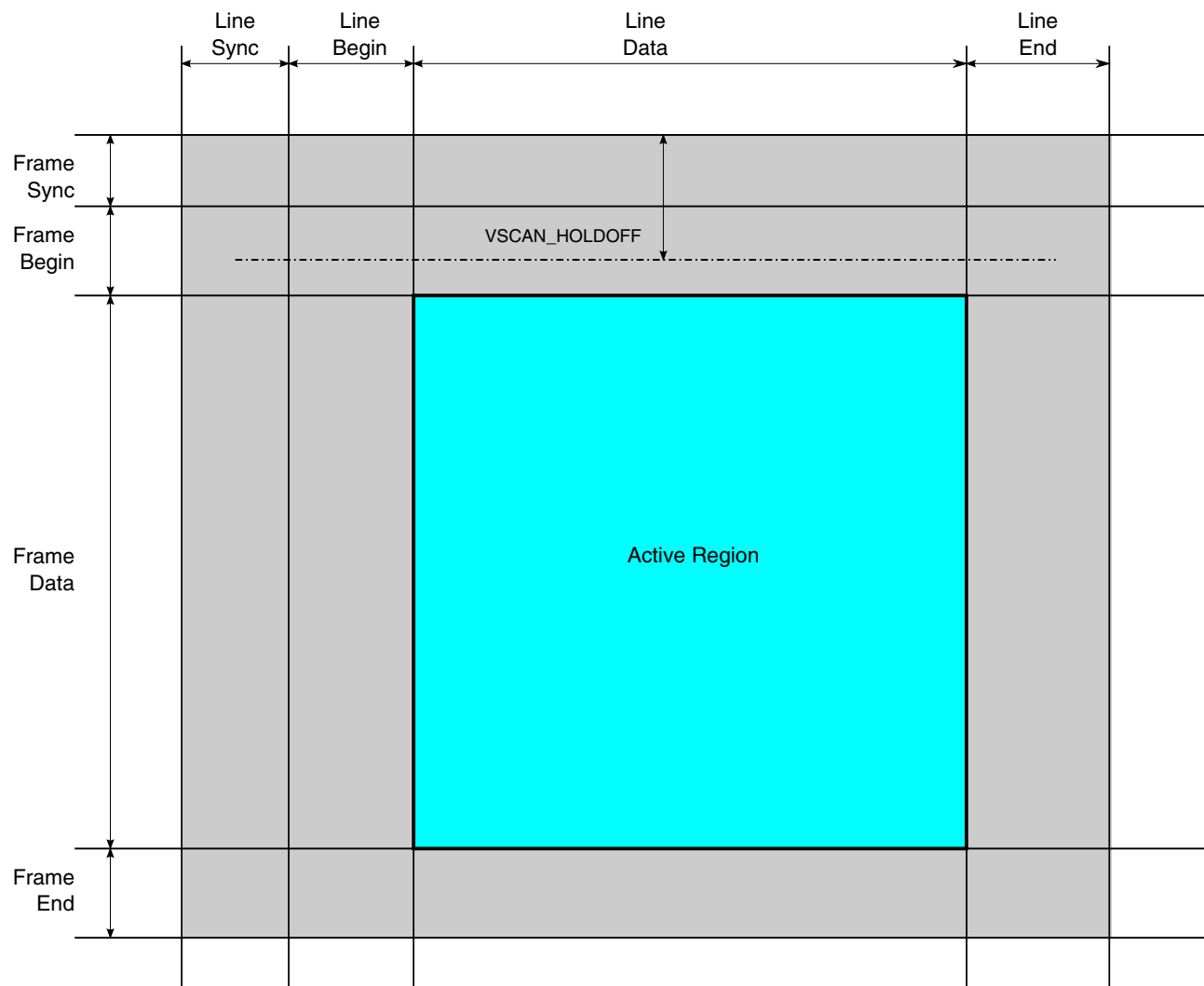
The EPDC provides a simple primary set of registers to define TFT line and frame (refresh) timing.

In addition to these, there are also low-level timing and polarity registers to provide additional flexibility in dealing with future TFT architectures.

The following figure shows how the horizontal and vertical timing is defined in terms of the various timing parameters. When configuring the TFT timing, the following goals should be met:

- Arrive at a refresh rate that matches the waveform requirement (for example, 50 Hz for 50 Hz waveform). The refresh rate is defined as the time between each frame sync event and can be expressed as a multiple of the line timing and the total number of vertical lines.
- Meet the various timing requirements of the gate and source driver clock and control signals (the blanking period provide an infrastructure for controlling these).
- Stay below the maximum source-driver clock (usually referred to as CL) frequency.
- Arrive at a source driver clock frequency which in turn defines the exact pixel clock frequency. In most cases, E-INK<sup>TM</sup> provides the expected SDCLK frequency.
- Meet the line-timing (LT) requirements of the E-INK panels and their associated waveforms.

For proper waveform performance, it is critical to match the refresh frequency to that required by the waveform and associated panel. Deviation from this frequency results in short term inaccuracy of color and in the long term, complete loss of coherency between the EPDC's internal representation of color and the physical representation on the screen.



**Figure 22-2. Frame Timing**

The following equations define the timing of the panel scan frame (where LT = line-timing and FT = frame-timing):

$$LT = TPIXCLK (LS + LB + (HRES \times \left( \frac{\text{Ratio}}{\text{PIXELSPERSDCLK}} \right) + LE)$$

$$FT = LT \times (FS + FB + VRES + LE)$$

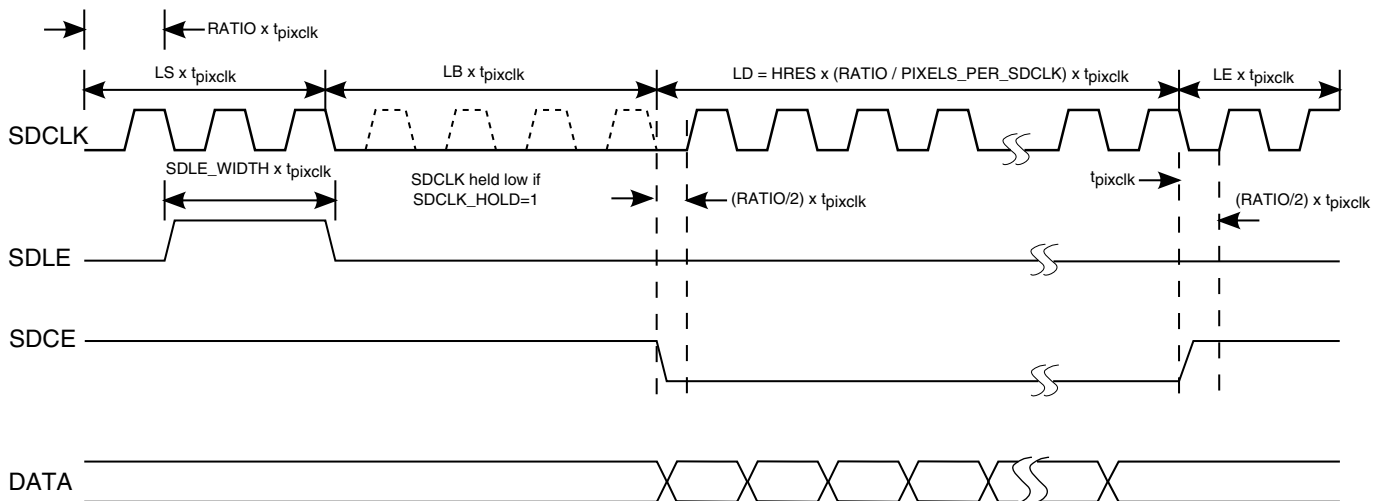
Ratio is defined between the internal PIXCLK and the external source driver clock SDCLK. The ratio is automatically determined by the EPDC according to [Table 22-2](#). A simple method to determine the ratio is as follows:

- When  $\text{DDR\_MODE} = 1$ ,  $\text{RATIO} = 4$
- When  $\text{DDR\_MODE} = 0$ 
  - If  $\text{PIXELS\_PER\_SDCLK} = 8$ ,  $\text{RATIO} = 4$
  - Else,  $\text{RATIO} = 2$

Typically it is recommended to seed these equations with an estimated (or provided) SDCLK frequency and adjust the various timings until the desired refresh rate is attained.

The following figure shows line timing for cases, where  $\text{DDR\_MODE} = 0$  (those cases where data is only sampled by the source driver on the rising edge of SDCLK).

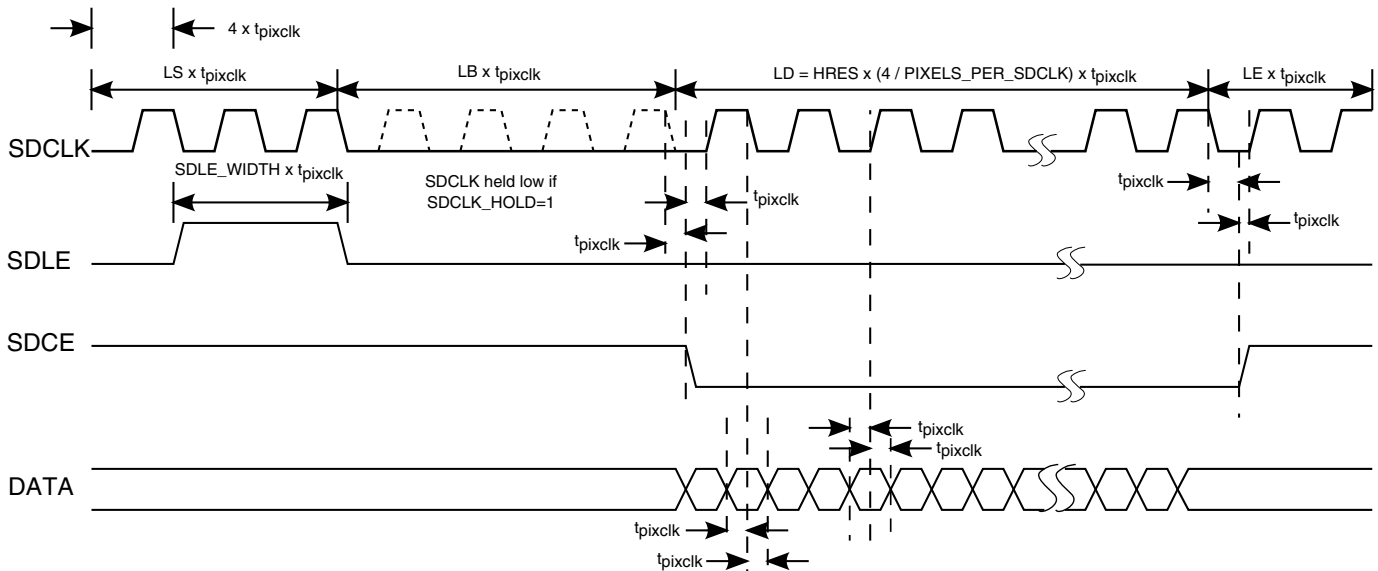
Depending on the mode (see [Table 22-2](#)), the ratio is a function of TFT pixel width and  $\text{PIXELS\_PER\_SDCLK}$ , so the timings are shown generically. For the purposes of this diagram,  $\text{RATIO}$  can either be 2 or 4.



**Figure 22-3. Line Timing ( $\text{DDR\_MODE} = 0$ )**

As per [Table 22-2](#), when  $\text{DDR\_MODE}=1$  (for example, for source drivers that require data to be driven on each edge of SDCLK), the  $\text{RATIO}$  is always set to 4. The timing for such cases is shown in the figure below.

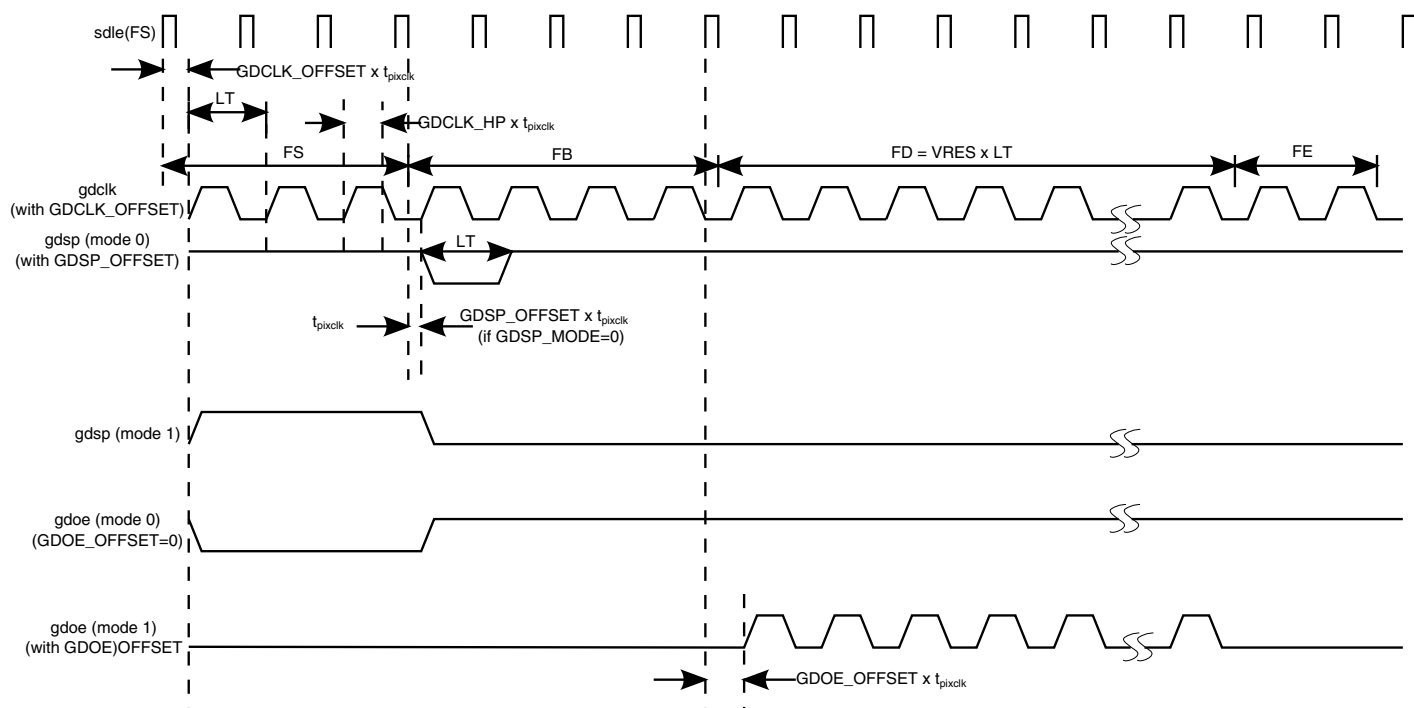
## Programming Model



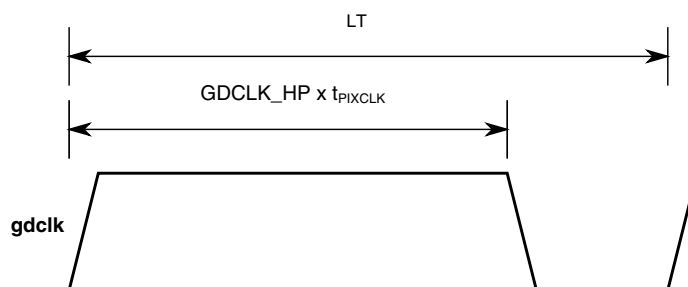
**Figure 22-4. Line Timing (DDR\_MODE = 1) - RATIO = 4**

Vertical timing or frame timing (FT) is always expressed in terms of line timing (LT). This is shown in the following figure, followed by a detailed view of the GDCLK timing (which has programmable duty cycle to allow programming of gate-on/gate-off time) in [Figure 22-6](#). The following important points should be noted in regard to gate driver timing:

- All gate driver timing signals are independently referenced to the LINE\_SYNC leading edge (this includes GDCLK\_OFFSET, GDSP\_OFFSET, and GDOE\_OFFSET).
- GDSP\_OFFSET is only supported for GDSP\_MODE = 1
- For GDOE\_MODE = 1, if GDOE is expected to be delayed relative to GDCLK, the GDOE\_OFFSET value should be set at a value appropriately greater than GDCLK\_OFFSET.
- All offset settings are programmed in terms of PIXCLK cycles.
- By default, if all offset values are set to 0, the gate driver signals are always one PIXCLK cycle delayed from the LINE\_SYNC leading edge.
- The LINE\_SYNC leading edge doesn't depend on the LINE\_SYNC\_WIDTH value. In cases where LINE\_SYNC\_WIDTH is set to some value that is less than LINE\_SYNC, a virtual leading edge which determines the line start time still exists.



### Figure 22-5. Frame Timing (Vertical)



### Figure 22-6. GDCLK Duty Cycle

An example calculation based on the 800 x 600 E-INK 6" panel with 50 Hz waveforms is shown be:

- $\text{PIXELS\_PER\_SDCLK} = 4$
- $\text{Ratio} = 2$  (2 bpp, 8-bit single-ended, SDR per [Table 22-2](#))
- $\text{tPIXCLK} = 17.64 \text{ MHz}$  (assuming available chip clock frequency)
- $\text{LS} = 20 \times \text{tPIXCLK}$
- $\text{LB} = 8 \times \text{tPIXCLK}$
- $\text{LD} = 800 \times (2/4) \times \text{tPIXCLK} = 400 \times \text{tPIXCLK}$
- $\text{LE} = 142 \times \text{tPIXCLK}$
- $\text{LT} = (20 + 8 + 400 + 142) \times \text{tPIXCLK} = 32.31293 \text{ uS}$

LINE\_SYNC\_WIDTH does not affect line timing. It should be set to the same value as LINE\_SYNC, unless LB and LE are short and it is used to shorten the width of SDLE whilst still maintaining the same line-timing.

It is important to meet the target line time of the panel which was 32.312 uS. Then the frame-rate can be calculated and given the vertical timings:

- $FS = 4 \times LT$
- $FB = 4 \times LT$
- $FD = 600 \times LT$
- $FE = 10 \times LT$
- $FT = (4 + 4 + 600 + 10) \times LT = 50.07665 \text{ Hz}$

In addition to this, E-INK panels require a particular duty cycle for the GDCLK signal which is specified as gate-on-time and gate-off-time. The sum of gate-off and gate-on should equal LT and the value of gate-on can be programmed through `HW_EPDC_TCE_TIMING2[GDCLK_HP]`, or GDCLK High-Pulse time.

### 22.3.4.4 Source Driver and Pixel Clock Configuration

The EPDC supports a number of source driver architectures. Each architecture forces a particular shift clock pixel rate.



In addition to this, the selection of the architecture requires a particular clock ratio between the internal pixel clock (pixclk) and external shift clock EPDC\_SDCLK. The table below outlines the various configurations and clock ratios. It also lists the relevant register settings for this configuration.

**Table 22-2. Interface Modes**

High-Level Mode		HW_EPDC_CTRL [DUAL_SCAN]	EPDC_DATA pins used	HW_EPDC_FORMAT [TFT_PIXEL_FORMAT]	HW_EPDC_TCE_CTRL [PIXELS_PER_SCLK]	HW_EPDC_TCE_CTRL [LVDS_MODE]	HW_EPDC_TCE_CTRL [DDR_MODE]	HW_EPDC_TCE_CTRL [SDDO_WIDTH]	Required PIXCLK (RATIO)
Single Scan	2bpp, 8-bit single-ended, SDR	0	[7:0]	2B[V]	FOUR	0	0	8BIT	SDCLK x 2
	2bpp, 16-bit single-ended, SDR	0	[15:0]	2B[V]	EIGHT	0	0	16BIT	SDCLK x 4
	2bpp, 8-bit, DDR (LVDS option)	0	[7:0],[15:8]	2B[V]	EIGHT	110	1	8BIT	SDCLK x 4
	4bpp, 8-bit single-ended, SDR	0	[7:0]	4B[V]	TWO	0	0	8BIT	SDCLK x 2
	4bpp, 16-bit, single-ended SDR	0	[15:0]	4B[V]	FOUR	0	0	16BIT	SDCLK x 2
	4bpp, 8-bit, DDR (LVDS option)	0	[7:0],[15:8]	4B[V]	FOUR	110	1	8BIT	SDCLK x 4
	4bpp, 16-bit, single-ended, DDR	0	[15:0]	4B[V]	EIGHT	0	1	16BIT	SDCLK x 4

*Table continues on the next page...*

**Table 22-2. Interface Modes (continued)**

High-Level Mode		HW_EPDC_CTRL [DUAL_SCAN]	EPDC_DATA pins used	HW_EPDC_FORMAT [TFT_PIXEL_FORMAT]	HW_EPDC_TCE_CTRL [PIXELS_PER_SCLK]	HW_EPDC_TCE_CTRL [LVDS_MODE]	HW_EPDC_TCE_CTRL [DDR_MODE]	HW_EPDC_TCE_CTRL [SDDO_WIDTH]	Required PIXCLK (RATIO)
Dual Scan	2bpp, 8-bit single-ended, SDR	1	[7:0],[15:8]	2B[V]	FOUR	0	0	8BIT	SDCLK x 2
	2bpp, 8-bit, DDR	1	[7:0],[15:8]	2B[V]	EIGHT	0	1	8BIT	SDCLK x 4
	4bpp, 8-bit single-ended, SDR	1	[7:0],[15:8]	4B[V]	TWO	0	0	8BIT	SDCLK x 2
	4bpp, 8-bit, DDR	1	[7:0],[15:8]	4B[V]	FOUR	0	1	8BIT	SDCLK x 4

### 22.3.4.5 Initializing the Display

Because of the nature of the EPD technology, a typical scenario would involve the EPDC's working buffer (WB) being maintained in system memory. In such power states (where memory is maintained), it is not required to initialize the display (even if the EPD panel had been powered off for example).

In low-power modes where system memory is not maintained, there are two possible methods to initialize the display.

The first method includes initialization from an unknown state and the second mode simply requires software to load the last state of the WB into memory and configure the EPDC to point to it.

#### 22.3.4.5.1 Reset/Clocks and Buffer Preparation

In cases in which the user wants to initialize the screen to a new known state, the following sequence should be followed. Assume that all relevant display power supplies are active.

First, the user must complete a soft reset sequence which includes enabling the main EPDC block-level clock gate. This sequence is described in the example code below. Setting SFTRST enables the CLKGATE. In order to correctly cycle reset values through

pipeline registers, the CLKGATE bit stays asserted for some finite amount of time before it sets. After this time, it is safe to clear both the SFTRST and CLKGATE. The example below shows a typical use of these registers.

```
EPDC_CTRL_SET(BM_EPDC_CTRL_SFTRST);
while (!EPDC_CTRL.B.CLKGATE);
EPDC_CTRL_CLR(BM_EPDC_CTRL_SFTRST | BM_EPDC_CTRL_CLKGATE);
while (EPDC_CTRL_RD() & (BM_EPDC_CTRL_SFTRST | BM_EPDC_CTRL_CLKGATE));
```

Before any display update is performed, the waveform address and working buffer (WB) pointers must be set (both must be aligned to a 64-bit double long word address):

- EPDC\_WVADDR: Must point to the base address of the waveform data (managed by the i.MX driver)
- EPDC\_WB\_ADDR: An area of memory must be assigned for the EPDC's working buffer (WB). Once assigned, this memory space should be reserved purely for the use of the EPDC. The requirements for the memory allocation are described in [Memory Requirements](#).

After this, the various panel configuration parameters must be set including resolutions, source and gate driver formats, TFT and buffer pixel formats.

#### 22.3.4.5.2 Performing an Initialization Display Update

Once all the panel timing and format parameters are defined, a command sequence must be sent to the EPDC with the properties found here.

- Update size must be full screen resolution
- HW\_EPDC\_UPD\_CTRL[UPDATE\_MODE] must be set to FULL
- HW\_EPDC\_UPD\_CTRL[WAVEFORM\_MODE] must be set to the INIT waveform (typically 0x00)
- HW\_EPDC\_UPD\_CTRL[USE\_FIXED] must be set
- HW\_EPDC\_UPD\_FIXED[FIXNP\_EN], FIXCP\_EN must be set to 1, and FIXNP and FIXCP must be set to 0xFF

The use of HW\_EPDC\_UPD\_FIXED allows the EPDC working buffer to be primed to a state that matches the result of the initialization waveform.

If the application loads the previous WB of the EPDC, these steps are not necessary.

### 22.3.5 Dual-Scan Configuration

The EPDC supports ultra-high resolution/refresh rate EPD panels.

At certain resolutions/refresh rates, there exist electrical limitations of the TFT devices and TFT driver ICs such as source drivers. In order to support driving such displays, the EPDC supports the dual-scan driver method.

The specific method involves driving a TFT configuration where there are two sets of source drivers. All source driver control signals are expected to be shared across all source drivers (except the SDCE signals unless a unified CE is presented by the panel).

In this mode, EPDC\_DATA[7:0] is reserved to drive the upper half of the display and EPDC\_DATA[15:8] is reserved to drive the lower half of the panel. The update and working buffers do not have to be aware of the dual-scan operation because the MBM knows how to access the WB for refresh operations based on the settings of the EPDC\_TCE\_CTRL[SCAN\_DIR\_0,1]. The EPDC supports four scan configurations for dual-scan which are shown in the following figures. Each figure shows an example configuration (that is to say, the EPDC is not limited to the support of only 2 source drivers per side).

The dual-scan drive method allows the SDCLK frequency to be halved relative to a single-scan method for the same resolution and refresh rate. In addition to this, each TFT is only charged in half frequency (for any given SDCLK cycle, twice as many pixels are being driven per frame scan).

Because all 16 DATA signals must be utilized for dual-scan, there is no support for the LVDS\_MODE. The supported source-driver modes for dual scan are shown in [Table 22-2](#). For the 2bpp DDR mode, the EPDC is capable of generating a total of 16 pixels per SDCLK cycle.

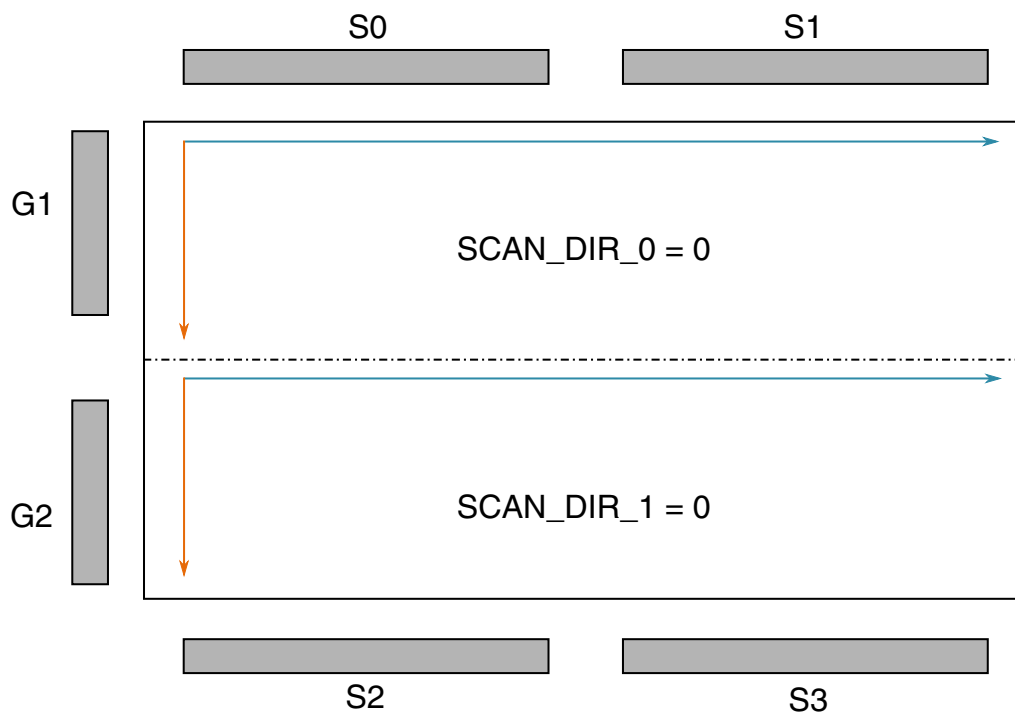


Figure 22-7. Dual-Scan Method (Scan = 00)

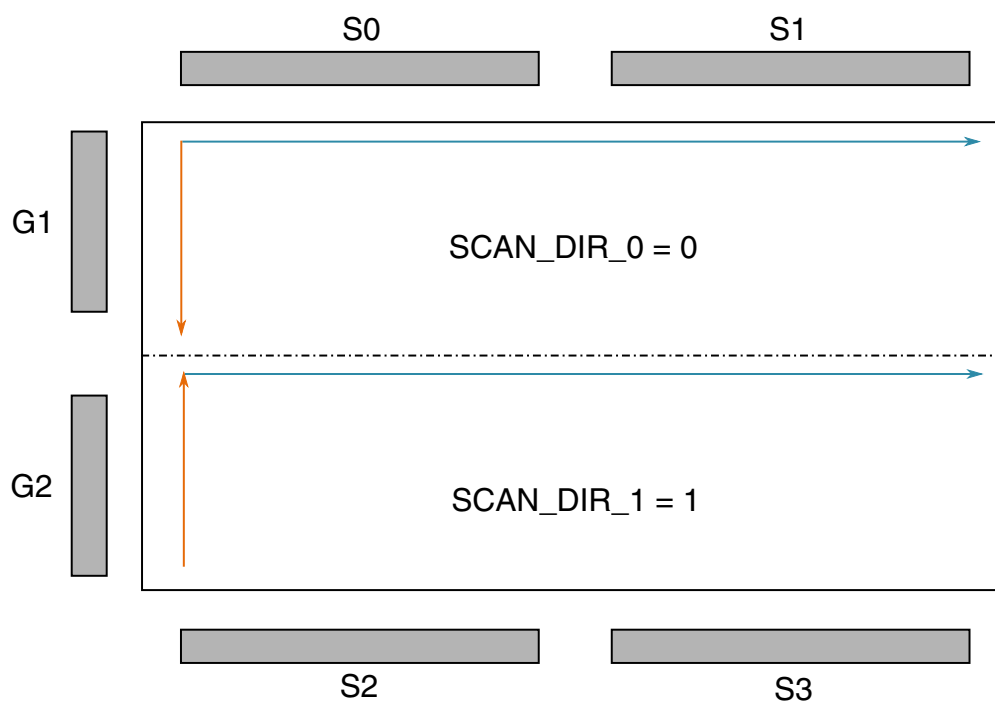
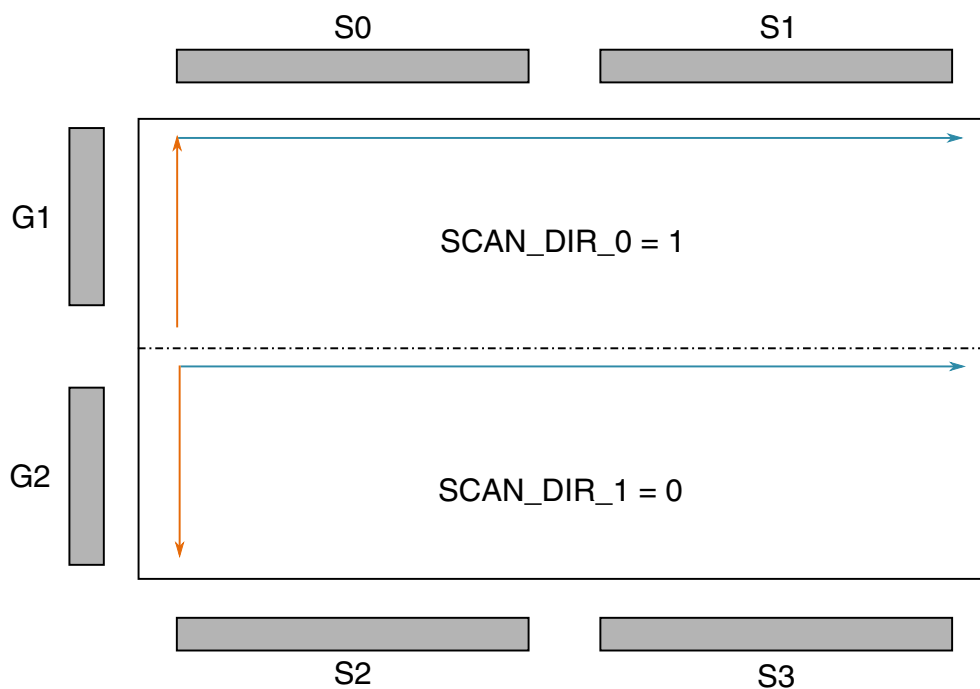
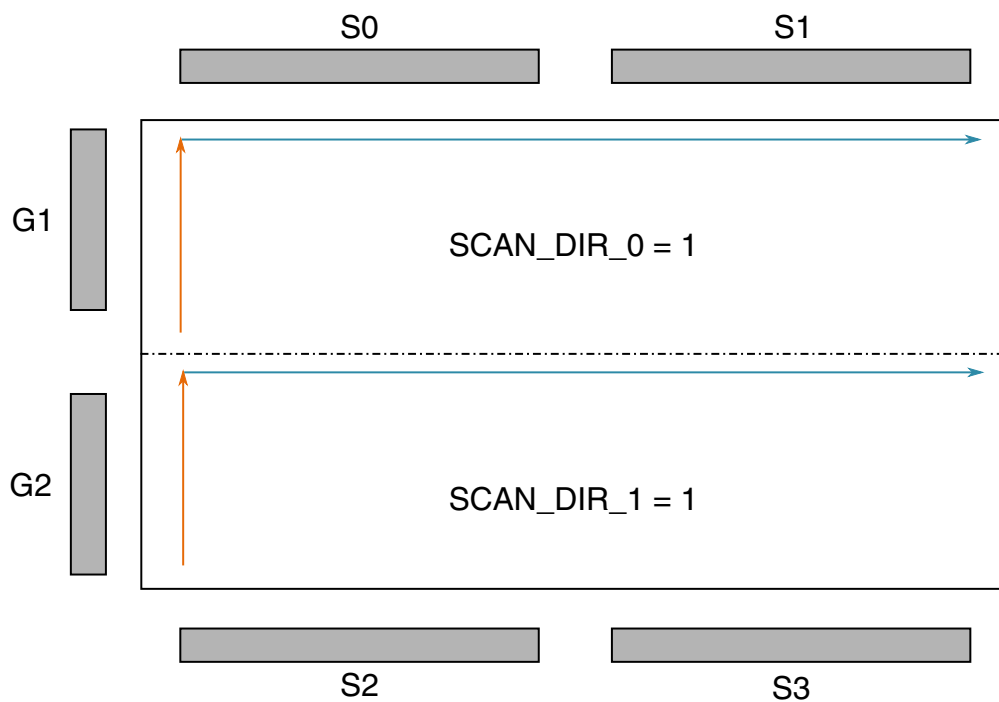


Figure 22-8. Dual-Scan Method (Scan = 01)



**Figure 22-9. Dual-Scan Method (Scan = 10)**



**Figure 22-10. Dual-Scan Method (Scan = 11)**

### 22.3.6 Update Buffer Analysis Functions

EPDC can perform several analyses on the update buffer.

The information collected during processing will be reported to the register after update buffer processing is complete.

- **Collision Rectangle Detection**

For a collided update buffer, this function will report a minimal rectangle that can cover all collided pixels.

When a collision occurs, it's possible that not all, but only part of that update buffer generate a collision, so only the pixels inside a minimal collision rectangle need to be re-submitted.

Details of minimal collision rectangle reported in UPD\_COL\_CORD/  
UPD\_COL\_SIZE.

- **Grey Level Detection (histogram)**

This function reports the minimal grey level that covers all pixels and which must be updated. This histogram differs from that of PXP; here we ignore those pixels which need no update, while the PXP histogram is calculated on the whole update buffer. The histogram feature is capable of reporting a histogram for 1, 2, 4, 8, or 16 grey levels. The valid grey levels value should be programmed into the HIST(1/2/4/8/16)\_PARAM registers before using this functionality.

- **Dry-Run Mode**

Dry-Run mode allows the user to run an update for the purpose of obtaining information without performing any real action. This mode is enabled by setting UPD\_CTRL[DRY\_RUN] = 1 when writing UPD\_CTRL register .

In this mode, Update Buffer will be read and checked against the current Working Buffer. As with a normal update, an interrupt will be generated when the Working Buffer processing completes.

Several pieces of information about the update can then be acquired from the EPDC registers: whether a collision occurred, the minimal collision area, and histogram data.. However, the result (Next Pixel, LUT info) will not be written back into the Working Buffer for further panel scan operation, nor will the LUT waveform be loaded.

The LUT resource specified by `UPD_CTRL[LUT_SEL]` is not used here; the user can start this dry-run update when no LUTs are available.

### 22.3.7 Waveform Mode Selection (AUTOWV)

This function is provided to support waveform selection based on the grey level (histogram) information collected during the update buffer processing stage.

Then AUTOWV LUT is provided to hold a mapping between the grey level of NP(pixel value from update buffer) and optimal waveform mode. This LUT mapping should be programmed before using this function by writing into the AUTOWV\_LUT register with `AUTOWV_LUT[ADDR]="grey level"`, `AUTOWV_LUT[DATA]="waveform mode"` for each possible grey level reported by the EPDC histogram.

After AUTOWV\_LUT is programmed, AUTOWV mode can be enabled by setting `UPD_CTRL[AUTOWV]=1`. When enabling AUTOWV, the waveform mode written into `UPD_CTRL[WAVEFORM_MODE]` will be ignored and the loading waveform will initially be put on hold.

After the Update Buffer processing completes, EPDC reports the minimal grey level, retrieves the mapped waveform mode from AUTOWV LUT, and writes it back into `UPD_CTRL[WAVEFORM_MODE]`.

If the user also sets `UPD_CTRL[DRY_RUN]`, EPDC will be finished at this point.

Otherwise, EPDC continues with the following, depending upon the `UPD_CTRL[AUTOWV_PAUSE]` setting:

1. `UPD_CTRL[AUTOWV_PAUSE] = 0, AUTO MODE)`

EPDC will begin waveform loading immediately without any software interaction

2. `UPD_CTRL[AUTOWV_PAUSE] = 1, MANUAL MODE)`

EPDC waits for software to decide whether the selected waveform is ok, then if necessary software can use some other criteria to select a waveform mode. After software selects the waveform mode, it writes again into `UPD_CTRL` with final waveform mode, and waveform loading then panel scan will start.

#### NOTE

AUTOWV\_PAUSE is extended for general use so that it can be used independent of AUTOWV feature. Regardless of whether AUTOWV is enabled or not, the user can send updates with `AUTOWV_PAUSE=1`. EPDC will stop after WB processing



and pause before LUT loading and panel scan. Software can explicitly do that latter by writing to UPD\_CTRL with AUTOWV\_PAUSE=0. EPDC will ignore the Update Buffer for this time, and kick off the LUT loading and panel scanning. This feature allows SW modification to the working buffer after EPDC hardware has finished processing it.

### 22.3.8 Panel Interface Generator (Pigeon Mode)

There are seventeen panel interface signal outputs (DATA/SDCLK not included), each of them with dedicated timing purpose as default. This is called "Legacy Mode".

Pigeon Mode is a timing mode which can be independently enabled on any of the seventeen timing signals, with a unified flexible configuration. Signals within pigeon mode can be programmed into any supported signals, and are interchangeable.

The following are the legacy timing signals which support pigeon mode:

- PIGEON[00] - SDCE0
- PIGEON[01] - SDCE1
- PIGEON[02] - SDCE2
- PIGEON[03] - SDCE3
- PIGEON[04] - SDCE4
- PIGEON[05] - SDCE5
- PIGEON[06] - SDCE6
- PIGEON[07] - SDCE7
- PIGEON[08] - SDCE8
- PIGEON[09] - SDCE9
- PIGEON[10] - SDOE
- PIGEON[11] - SDL3
- PIGEON[12] - SDOEZ
- PIGEON[13] - SDOED
- PIGEON[14] - GDSP
- PIGEON[15] - GDOE

#### Working Theory

Each pigeon signal has one local counter with a configurable start point and incremental condition. It will be compared to configuration register value for signal assertion/de-assertion control, plus delta offset for data alignment and other options like polarity/logic operation. A detailed running scenario is as follows:

1. Start local counter on the MASK rising edge (reference point/start point)

2. Increment on event selected through INC\_SEL
3. Count and match SET\_CNT: assert signal, reset counter  
( SET\_CNT==0 means assert immediately on MASK's rising edge )
4. Count and match CLR\_CNT: de-assert signal, stop counter  
(CLR\_CNT==0 means de-assert on MASK's falling edge)

### NOTE

When local counter is running, further changes to MASK are not cared unless CLR\_CNT==0

MASK - start point for local counter

Created using any combination of below options (ANDed)

1. STATE\_MASK = (FS|FB|FD|FE) AND (LS|LB|LD|LE)

8 bits to select in which vertical/horizontal state your counter starts ticking. This is the most common use-case because timing signals generally relate to scan states.

For example, for a line timing signal start on Line Begin phase during Frame Begin/ Frame Data lines, use the configuration below:

**Table 22-3. STATE\_MASK combinations**

STATE_MASK	LS	LB =1	LD	LE
FS				
FB =1		X		
FD =1		X		
FE				

2. MASK\_CNT/MASK\_CNT\_SEL(global counter)

Sometimes a more accurate reference point is required, such as " line 20 in a frame", or "line 12 in Frame Begin state" or "pixel 23 in LD phase". For such use-cases, several Global Counters shared by all pigeons are provided. Global counter type (line counter/frame counter/state counter, etc.) is selected using MASK\_CNT\_SEL, when global counter matches the MASK\_CNT value. The pigeon local counter will start ticking.

3. Use another pigeon signal as mask (SIG\_LOGIC=MASK)

For some tightly coupled signals it is possible to use one as a reference to generate another.

INC\_SEL- select local counter tick event

1. pclk - pixel clk
2. line - line start pulse
3. frame - frame start pulse
4. another - another pigeon signal

OFFSET- offset on pclk basis

Some signals need to come out slightly earlier or later than programmed. For example, the CE type signal usually aligns with the Line Data phase, but some panels need it as one cycle pulse before Line Data. The user can set OFFSET to a negative value to achieve this.

Global Counters (selectable through MASK\_CNT\_SEL)

- HCNT / VCNT

normal pclk counter / line counter

- HSTATE\_CNT / VSTATE\_CNT

similar to above, but reset when state changes

- HSTATE\_CYCLE / VSTATE\_CYCLE

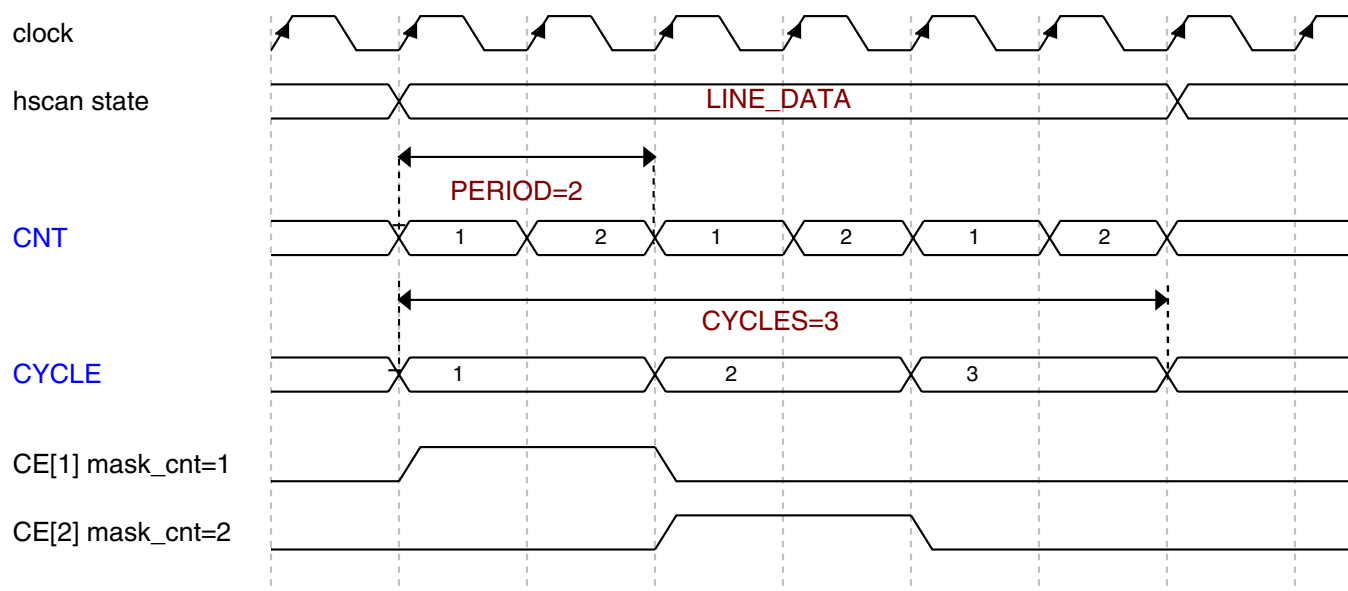
(see figure below for definition of CYCLE/PERIOD/CNT)

Some panels have multiple Gate Drivers/Source Drivers, so Frame Data / Line Data state may be further split to match each driver, and signals such as CE[n] are only valid during part n of Line Data. For such signals, use HW\_EPDC\_PIGEON\_CTRL.\*\_PERIOD to specify PERIOD where CNT is reset and CYCLE is included. Then the user can select CYCLE as MASK\_CNT to generate mask for CE[n].

- FRAME\_CNT / FRAME\_CYCLE (only for frame-crossing signals)

frame cycle counter doesn't have a reset condition; use

HW\_EPDC\_PIGEON\_CTRL1.FRAME\_CNT\_CYCLES to reset it.



**Figure 22-11. Definition of CNT, CYCLE, PERIOD**

The following are register settings for the figure above:

- HW\_EPDC\_PIGEON\_CTRL0.LD\_PERIOD=2
- MASK\_CNT\_SEL = 1 // HSTATE\_CYCLE
- MASK\_CNT = 1 // CE[1]
- MASK\_CNT = 2 // CE[2]

## 22.3.9 Display Update Programming

The EPDC is designed to communicate with the kernel driver through the interrupts and status registers.

The driver entry is always assumed to occur as a result of an interrupt.

### 22.3.9.1 Initiating a Display Update

The typical flow for performing a display update involves the following:

- EPDC\_STATUS[WB\_BUSY] must be 0. The EPDC cannot process new updates if it is currently processing an update.
- EPDC\_STATUS[LUTS\_BUSY] must be 0. The EPDC cannot accept new updates if all LUT resources are currently active.

Assuming both conditions above are met, an update request is programmed in the following manner (it is important to note that the last step must be writing to EPDC\_UPD\_CTRL which initiates the display update).

The following steps can be performed before WB\_BUSY and LUTS\_BUSY are clear:

- At this time it is assumed that the correct temperature is selected and written to EPDC\_TEMP.
- EPDC\_UPD\_ADDR must be set to the 8-bit buffer for the update **if stride feature not enabled (when enabled stride feature, no requirement on align )**. This buffer must be aligned to a 64-bit word address. In addition, each line address must begin at a 64-bit word address. Both criteria can be met by utilizing the PXP pixel processing engine which always aligns lines at a 64-bit raster (see [Memory Requirements](#) for details).
- EPDC\_UPD\_CORD defines the insertion co-ordinate into the panel resolution.
- EPDC\_UPD\_SIZE defines the dimension of the update (pixels).
- EPDC\_UPD\_FIXED is used for panel initialization or rectangular fill operations.

Writing the EPDC\_UPD\_CTRL register initiates a display update:

- USE\_FIXED-This field is set when EPDC\_UPD\_FIXED is used.
- LUT\_SEL-The driver should read EPDC\_STATUS\_NEXLUT[NEXT\_LUT] (qualified by NEXT\_LUT\_VALID) to select an available LUT to assign this update.
- WAVEFORM\_MODE selects the E-INK waveform mode (for example, INIT, DU, GC16 and so on). The number corresponds to the waveform number in the waveform specification document.
- UPDATE\_MODE selects one of two enumerated updated modes.
  - FULL-In this mode, all pixels defined in the update rectangle have the waveform applied.
  - PARTIAL-In this mode, the EPDC shall only apply the waveform to pixels which are changing, otherwise the EPDC\_FORMAT[DEFAULT\_TFT\_PIXEL] is applied to the pixels which are not changing.

### 22.3.9.2 Update Processing and Collisions

After the update is initiated (by writing EPDC\_UPD\_CTRL), the EPDC performs update-buffer processing. This involves processing this update region into its working buffer (WB).

During this time, the EPDC also performs collision detection. At the end of the WB processing, the EPDC asserts the WB\_CMPLT\_IRQ interrupt. If a collision is detected, the LUT\_COL\_IRQ interrupt status bit will also be asserted.

The EPDC performs image updates to the EPD by applying waveforms to individual pixels. Waveforms are applied to groups of pixels (ranging from a single pixel to the entire display). Each individual update request is bound to a rectangle. Each pixel within this rectangle has a waveform applied to it using the waveform and update mode specified in EPDC\_UPD\_CTRL. After this process is initiated, a waveform takes an amount of time to complete which is usually determined by the specifications of the waveform mode. For proper display operation and optimum performance, a waveform must not be interrupted once it has begun.

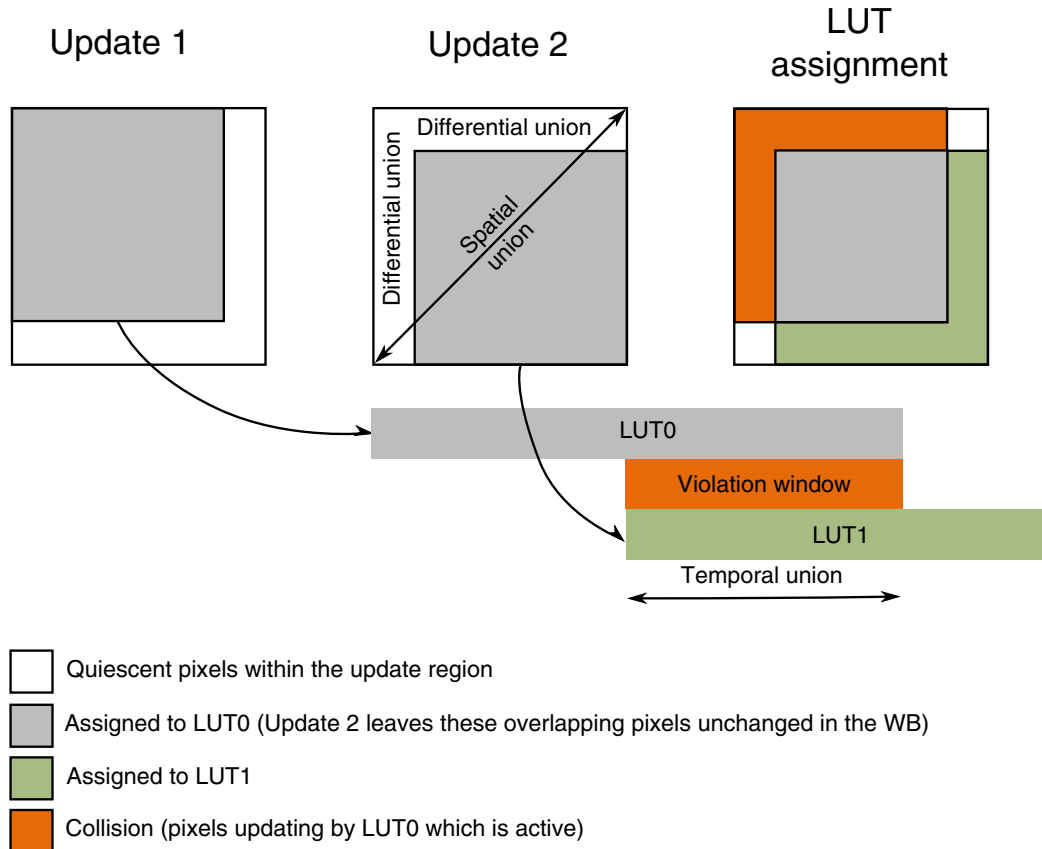
A collision is defined simply as the interruption of the application of waveform to a pixel or group of pixels. As such, it must be avoided. Because the waveforms take a finite amount of time to complete, the EPDC provides a mechanism to automatically manage collisions, which allows the application to perform multiple updates resulting in a richer user experience without having to be cognizant of the panel and waveform characteristics.

Depending on the update mode (FULL or PARTIAL) a collision is defined as follows:

- For a full-update mode, a collision is defined as the temporal and spatial union of rectangles pertaining to active unique update requests. An active update is one which is still in the process of being updated (that is, a waveform is being applied).
- For a partial-update mode, a collision is defined as the temporal, spatial and differential union of pixels pertaining to unique update requests which are active. The difference between a full-updated and a partial-updated collision is that in the latter case, the collision state occurs only if the next-pixel states differ between the updates. Thus, two partial-update rectangles may intersect, but if all of the pixels in the intersecting update regions are being updated to the same gray-state, the original update and associated LUT/waveform are not disturbed and the intersection does not constitute a collision.
- It should be noted that collisions may be a one-to-many effect. In the case of partial updates, there may be a spatial and temporal union of rectangles with no differential pixel-level union. A subsequent update however might form such a pixel-level differential union to all active updates. Since each update pertains to an active LUT, these many collided updates are referred to as victim LUTs. The update/LUT that caused the collision is referred to as the aggressor LUT.

An example of a partial-update mode collision is shown in the following figure. Each update encompasses the outer rectangle which meets the criteria of the spatial union. The red areas show the pixels which are currently transitioning to gray (from update 1) but need to transition to white (in update 2). These pixels meet the criteria of the differential union and lastly in the temporal domain, the period of time where LUT0 and LUT1 overlap is the temporal union. The pixels that did not change (grey and green areas in the LUT assignment box) are not part of the collision area. In the case of the gray

overlapping pixels, LUT0 can continue to update. In the case of the white pixels (which were previously inactive) a new LUT (if available) can be assigned) and forms the green region. Note that a subsequent update could potentially collide with the green pixels in update 2.



**Figure 22-12. Partial Update Collision Handling**

### NOTE

For FULL update modes, a collision occurs at any intersection of active rectangles/LUTs because in FULL update mode, all pixels within the update rectangle have the waveform applied (even if pixel values are not changing).

It can be seen that when collision is detected (during the WB update process), the EPDC will "block" the collided pixels being updated, and any pixels which are not collided will be assigned to the requested LUT process. When a collision occurs, the EPDC will raise the LUT\_COL\_IRQ interrupt status bit (this will always happen in conjunction with WB\_CMPLT\_IRQ).

Because the driver is always entered via an interrupt, when the EPDC interrupt fires, driver should not only check for WB\_CMPLT\_IRQ but also for the LUT\_COL\_IRQ status bit within the EPDC\_IRQ register. If LUT\_COL\_IRQ is set, this means that a

collision has occurred. If this is the case the driver should check the value of the EPDC\_STATUS\_COL register. This register contains one bit for each LUT that has been collided with, that is, it holds the vector of the victim LUTs as a result of the last update LUT. Since the driver knows the value of the LUT of the last update, it can store this update LUT value along with this associated victim LUT vector in a collision list.

Because the EPDC can drive up to 16 concurrent updates (which can be overlapping when using PARTIAL mode updates), a unique interrupt status bit is provided per LUT so that when an update completes (physically completes the waveform), the SW driver can know which LUT completed. In addition to the LUT completion interrupts (which are mapped into the EPDC\_IRQ[LUTn\_CMPLT\_IRQ] interrupt status bits), the current state of each LUT can also be read from the EPDC\_STATUS\_LUTS status register.

The driver can use the LUT status interrupts in conjunction with EPDC\_STATUS\_LUTS to perform regular checks on the completion of victim LUTs against the vectors stored for each colliding LUT in the collision list. The driver may also choose to serialize updates such that it might wait until all victim LUTs have completed before re-issuing the colliding update.

The figure below shows a more extended sequence of a collision. In this case there are a series of updates coming from the application layer. They are A, B, C and D. Updates A, B, and C are set without any collision. When update D is requested a collision is detected. Using the interrupt status collision registers, the SW-driver stores D in a collision-list. For that list entry it also store B which is the victim LUT. It uses this information to check when a LUT completes to see if the victim LUTs for any update in the collision list have completed. If they have completed (through a LUT completion interrupt), the SW-driver then sends this update (D in this case) again to the EPDC. It should be noted that from the application perspective it appears that all updates A, B, C and D have been accepted, that is, the collision handling is completely hidden from user-space. Using the collision detection and LUT status interrupts, the application (in conjunction with the driver) could also choose to completely serialize the requests (such that between the first and second D update, no other update is accepted).



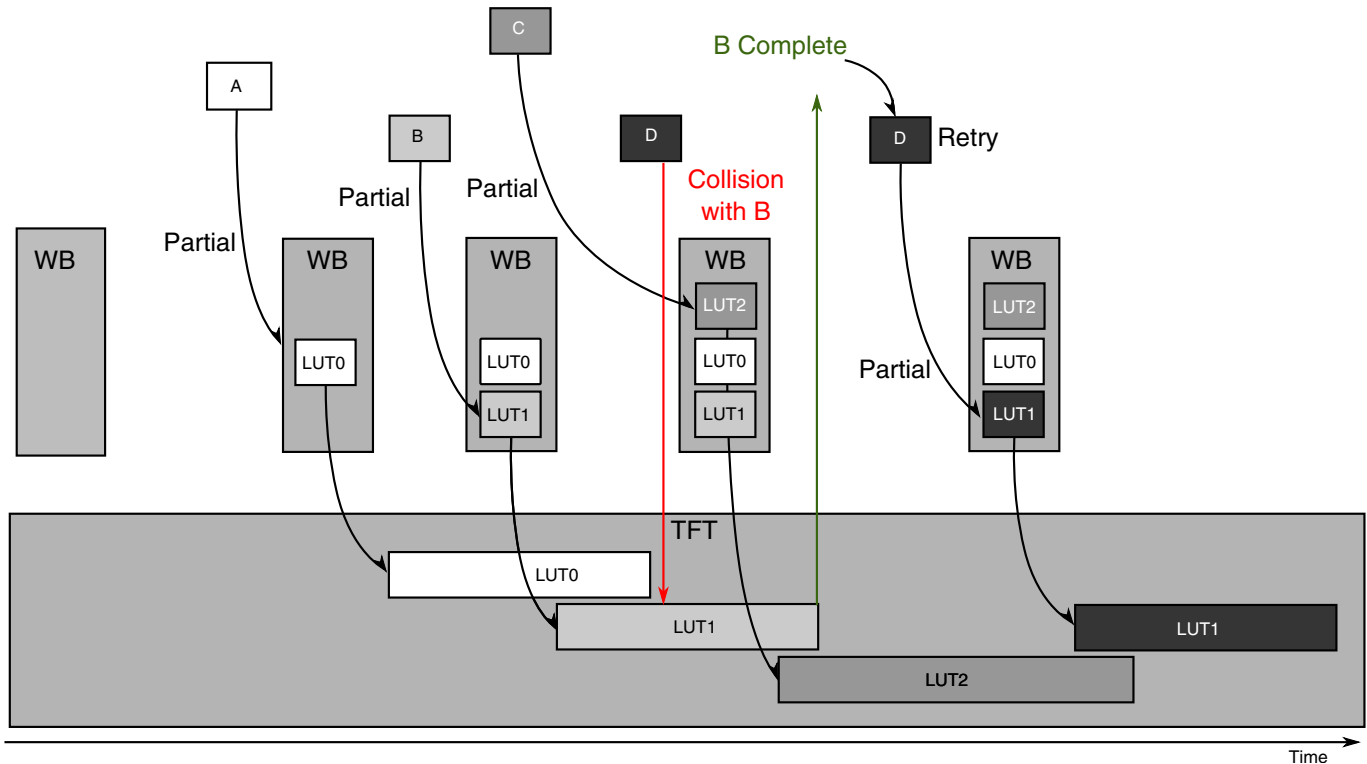


Figure 22-13. Collision Sequence

### 22.3.9.3 Multiple Update Flow

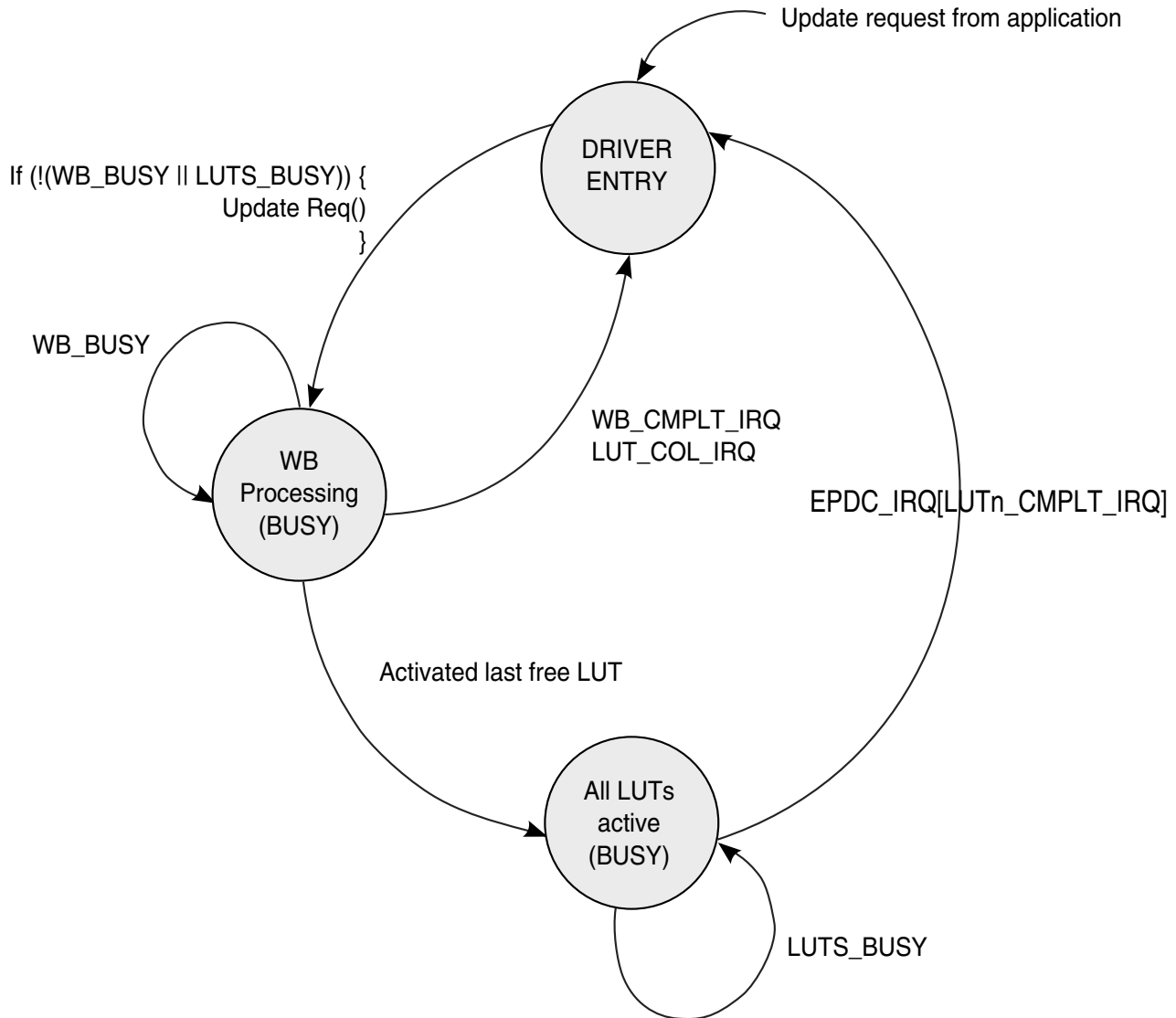
In most applications, such as e-book readers, besides full-screen page turns, user interfaces require multiple updates to be performed.

Examples of this include cursors, pen input, drop-down menus (with highlighting) and other movable graphical objects. Each update takes such a limited amount of time to complete (up to 1 second depending on the waveform mode) that it would be too prohibitive to only process updates serially.

As described in [Initiating a Display Update](#), there are two conditions that must be met before the SW driver can issue a new update request:

- The Working buffer (WB) update process must be complete.
- There must be an available LUT.

Assuming a free LUT is available, the EPDC is designed to have the capability to start new updates at each TFT frame scan (even being able to commence multiple updates within a frame scan blanking period). The general flow for sequential updates (concurrent or serialized) is shown in the following figure.



**Figure 22-14. Driver Update Flow**

This flow assumes that the driver is entered via interrupts. The two busy states are shown to illustrate the conditions under which the driver will not issue new updates. In general, the driver should be throttled by the WB\_CMPLT\_IRQ interrupt. Because the EPDC can support up to 64 concurrent updates, it is possible to re-enter the update-request (show as Update\_Req() in the figure above) any-time that the WB is not busy (even though LUTs are active and the display is being updated). It is possible that when there are 63 LUTs currently active and a new update is sent, there will be a WB\_CMPLT\_IRQ interrupt issued, but the driver at that time would be in a busy state (because LUTS\_BUSY is signaled). The driver will be re-entered on the next LUTn\_CMPLT\_IRQ interrupt during which time it can issue a new update (since WB\_BUSY and LUTS\_BUSY status bits are low).

The update-processing process (which is controlled by the MBM module) is asynchronous to the actual TFT refresh operations (which carry out the physical screen updates), so there is no particular relationship between the WB\_BUSY status and the state of the screen update. In contrast, the LUTn\_CMPLT\_IRQ interrupt status and EPDC\_STATUS\_LUTS provide the status of the physical waveform update. This means that when LUTn\_CMPLT\_IRQ interrupt fires, it means that the last physical TFT frame scan just completed for that update/LUT.

The EPDC implements proprietary methods to efficiently process the WB such that the WB processing time (the time from the SW driver writing EPDC\_UPD\_CTRL to EPDC\_IRQ[WB\_CMPLT\_IRQ] fires), is dependent on the size of the update rectangle. This allows smaller updates to be processed faster. In most EPD applications, this is ideal because it's typically smaller sprites that require the fastest performance. In addition, the EPDC implements methods to allow store pending updates and activate them on the next available frame-scan. For example, if the TFT scanning is currently in the active region and during this time a series of small update requests are sent, the EPDC can begin all of those updates on the next available blanking period.

The figure below shows an example of two larger update requests, one to LUT4 and one to LUT5. These update requests are shown in the context of active LUTs and thus active frame scan times. The time when both WB\_BUSY and LUT\_BUSY is high are defined as "blocking" because no new updates can be accepted. It should also be noted that new updates are physically commenced by the EPDC on blanking periods. From a SW programming model perspective, the EPDC\_STATUS\_LUTS[LUTn\_STATUS] is activated immediately upon the update request being written to the EPDC. The physical frame-scanning for this particular update does not being commenced until the next available frame scan.

It should be noted that updates and WB processing can actually occur during the blanking period (or can begin in the active scan time and continue into the blanking period). For this reason, the EPDC provides some tuning controls via EPDC\_TCE\_CTRL[VSCAN\_HOLDOFF] such that more time can be given to finish processing update request and less time "locking down" the set of LUTs that will be activated for the upcoming active scan. The EPDC will start to pre-fill its refresh pixel FIFOs after the VSCAN\_HOLDOFF period. Its possible to tune this value to allow the minimum time for the pixel FIFO pre-fill operation (making sure that refresh under-runs do not occur). If a pixel-FIFO under-run occurs, EPDC\_IRQ[TCE\_UNDERRUN\_IRQ] will fire.

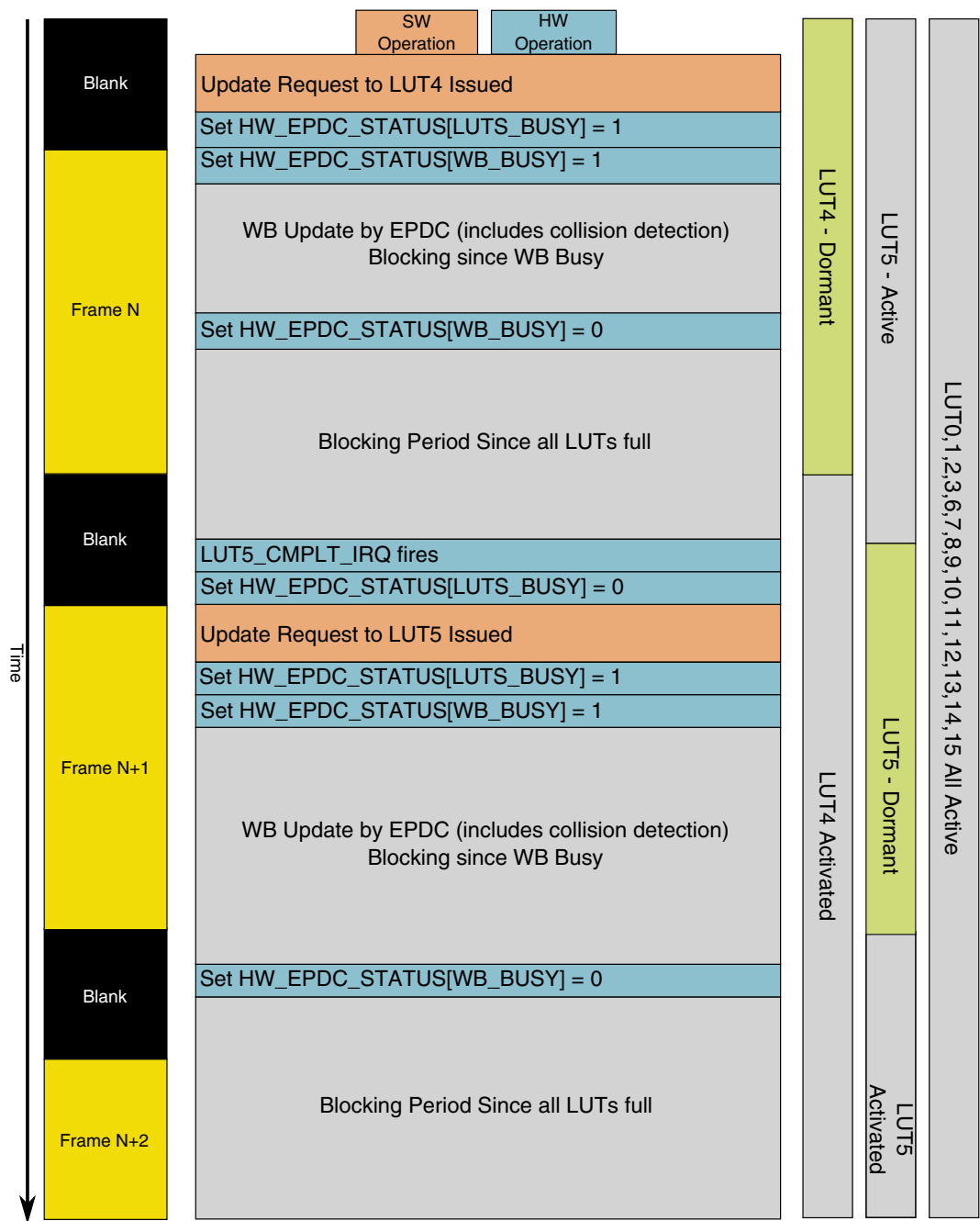


Figure 22-15. Temporal Flow of Update Processing

### 22.3.10 Architectural Clock Gating (Low Power Mode)

As with most modern IP, the EPDC contains low-level hardware controlled automatic clock gating throughout the design.

These clock gating elements typically gate clocks locally for individual or groups of sequential elements (flip flops) when these registers are not currently active.

This tier of clock gating is useful for reducing power during active operation of the EPDC.

Most e-paper applications utilizing EPDs involve the processor and display controller to be a low-power inactive state in most of the time. This is because EPDs only require refresh operations when the display content is being updated. As such, in between update operations, the EPDC provides a module level clock-gating feature that can be controlled by the SW driver as part of a higher-level power management solution.

The EPDC module can have all its clocks completely gated-off by the setting of `EPDC_CTRL[CLKGATE]`. There are a couple of considerations that the SW driver must adhere to before placing the EPDC into this low power state:

- All active LUTs (updates) must complete
- Any TFT operations must be complete (including any final blanking operations)

In order to maximize display update frequency (the ability accept new updates during each frame scan), the LUT completion interrupts fire on the end of the last frame's active scan period for that update. This means that the TCE still must complete the final `FRAME_END` blanking time to guarantee coherency of its state machines with the panel state (receiving the final LUT completion IRQ is not enough to determine that the EPDC can have all of its clocks shut down). Based on this, the EPDC provides the necessary interrupt and status bits to allow the driver to correctly perform the clock-gating operation by using the following method:

- Once the final `EPDC_IRQ[LUTn_CMPLT_IRQ]` is reached, the driver should enable the `TCE_IDLE` interrupt mask bit via `EPDC_IRQ_MASK[TCE_IDLE_IRQ_EN]`, unmask other interrupts and return.
- Once the final blanking time of the last LUT (`FRAME_END` time) is completed and the TCE reaches its idle state, the EPDC will assert the `EPDC_IRQ[TCE_IDLE_IRQ]` interrupt which will cause the driver to re-enter. At this point, the SW knows that the EPDC is in a completely idle state and can perform a set on `EPDC_CTRL[CLKGATE]`.

Alternative implementations of the driver could simply involve polling for the `EPDC_IRQ[TCE_IDLE_IRQ]` interrupt status bit (which operates regardless of the interrupt being masked or not) before setting the clock gate (this prevents two iterations of interrupt based driver-entry, but requires a wait loop in the interrupt handler which might be undesirable).

It should be noted that the time interval between the last LUTs `LUTn_CMPLT_IRQ` time and `TCE_IDLE_IRQ` interrupt is equal to the `FRAME_END` time.

When the application is ready to send more updates, the driver can immediately re-enable the EPDC by un-gating the module-level clock by clearing the EPDC\_CTRL[CLKGATE] field. This un-gating of the clock is instant and the driver is free to immediately send update requests.

## 22.3.11 Performance Tuning and Considerations

The EPDC is designed to meet the performance requirements of the most demanding e-paper applications. These performance targets must be met in the context of highly integrated SoCs.

In such a context, system memory is typically shared across multiple masters including the ARM platform. Because of this, there is no guarantee of latency from system memory. EPDC provides a number of mechanisms to deal with this as well as some ability to tune various parameters allowing the user to trade-off between update processing performance and refresh functionality.

### 22.3.11.1 Memory and Bus Bandwidth Requirements

The EPDC MBM module contains up to 6 internal AXI requesting functions (5 read and 1 write). These are arbitrated internally before being sent to the SoC memory system.

The EPDC interfaces to the SoC bus system via a dual-channel 64-bit AXI bus master port. The following table shows the various internal requesters and their properties. Note that there are two round-robin (RR) loops. Because there is only one write source, and the EPDC supports dual-channel AXI, there is no arbitration on the WB write operations.

**Table 22-4. EPDC Bus Requestor Profiles**

Operation	Priority	Burst Length (x8 Bytes)	Maximum Outstanding Transactions	FIFO size
Waveform LUT read	0	8	2	16x68
WB pixel read FIFO0 (refresh)	RR 1	8 or 16	8	256x64
WB pixel read FIFO1 (refresh)	RR 1	8 or 16	8	256x64
WB read for update processing	RR 2	8	2	16x64
WB write for update processing	N/A	8	2	16x64
UPD read for update processing	RR 2	8	2	16x64

As described in [Memory Requirements](#), it can be seen that all lines in a buffer are padded to the nearest 8 byte boundary. This is important to be noted for the bandwidth calculations. The maximum bandwidth scenario involves processing a full-screen update

while performing refresh operations, but refresh operations always have the highest priority. Based on this, the memory bandwidth required by the EPDC is a function of the refresh pixel rate which in turn is a function of resolution, frame-rate and blanking/active time.

The refresh bandwidth can be calculated as follows:

Active Time = Active time of the frame-scan time as a fraction (for example, 0.8)

Frame Rate = TFT frame refresh rate (Hz) (for example, 106 Hz)

Bandwidth (MB/s) =  $\text{roundup4}(\text{EPDC\_RES}[\text{HORIZONTAL}]) \times \text{EPDC\_RES}[\text{VERTICAL}] \times \text{Frame Rate} \times (1/\text{Active Time}) \times 2 \times (1/1024^2)$

For example, a panel with resolution 2048 x 1536 with 106 Hz refresh and estimated 80% active time, 20% blanking time, the refresh bandwidth is:

$\text{BW (MB/S)} = 2048 \times 1536 \times 106 \times (1/0.8) \times 2 \times (1/1024^2) = 795 \text{ MB/s}$

(Note that  $2048 \bmod 4 = 0$ .)

The WB requires 2 bytes per pixel.

Remaining bandwidth is used for other operations. Because update processing operations are not real-time (they do not necessarily have to occur at the refresh frame-rate), the time required to perform the update can be calculated as a function of the available bandwidth (actual bandwidth minus refresh bandwidth) and the update size.

### 22.3.11.2 Pixel Latency FIFOs

The EPDC contains two working buffer latency FIFOs which are used to load working buffer pixel data which is used by the TCE to perform the panel refresh operations.

These FIFOs are sized at 1024 pixels each in order to provide significant system memory latency tolerance. One pixel FIFO is dedicated for the main screen and the second is in dual-scan cases for the second half of the screen.

Under no circumstance should the EPDC pixel FIFOs reach an under-run condition. The EPDC provides an interrupt status bit to flag such a condition (TCE\_UNDERRUN\_IRQ). During development, this interrupt must be enabled. The pixel values stored in the FIFOs are used by the TCE to perform look-up operations (from the LUTs) to generate TFT voltage control pixels. If an under-run occurs, the result will be unknown data being used as the source for the look-ups, which can damage the panel.

### 22.3.11.3 Basic Watermarking Control

The EPDC provides a basic pixel pre-fetching control mechanism that is applied to the beginning of a new update (from an idle state). This control is provided by EPDC\_FIFOCTRL[FIFO\_INIT\_LEVEL].

The control allows the value to be set between 0-255, with a reset value of 128. Because the internal width of the FIFO holds 4-pixels per entry, the actual number of pixels is  $(\text{FIFO\_INIT\_LEVEL}+1) \times 4$ .

The EPDC uses this value to pre-fill the pixel latency FIFO(s) to this level before the TCE commences the refresh operations which drive the update to the panel. This control has no effect on the FIFO(s) when the EPDC is performing sequential frame-scans.

It must be noted that this value must be set less than  $(\text{EPDC\_RES}[\text{VERTICAL}] \times \text{EPDC\_RE}[\text{HORIZONTAL}]) / 4$ .

### 22.3.11.4 Update/Refresh Tuning (VSCAN\_HOLD OFF)

During active frame-scan time, the EPDC provides the ability to process and display new update requests, because all pixels in the WB pertaining to a particular update are bound to a single LUT.

Until all pixels pertaining to that update have been processed in the WB, its LUT cannot be activated. As shown in [Figure 22-15](#), the blanking period provides time to pre-load the pixel FIFOs and for completing WB processing updates and loading the associated frame of waveform data into LUT memories.

The EPDC\_TCE\_CTRL[VSCAN\_HOLD OFF] control field allows control over the vertical blanking period allotment for working buffer processing and pre-loading of the pixel FIFO. The time window defined by VSCAN\_HOLD OFF is allotted for working-buffer processing and LUT loading (of newly processed updates). The remainder of the time is allotted for pre-filling the pixel FIFO.

Note that aggressive use of VSCAN\_HOLD OFF can result in pixel FIFO under-run as the FIFO pre-fill time is compromised.

### 22.3.11.5 System-Level Arbitration Control

As previously mentioned, the TCE refresh operation is time critical.

Because the EPDC is designed to be integrated into a multimaster SoC, system memory is shared between multiple requesters in the system.



The EPDC provides a dynamic Quality of Service (QoS) priority elevation mechanism that can be leveraged at the SoC level.

The EPDC\_FIFOCTRL registers can be used to drive the `epdc_panic` output signal during times when the EPDC requests require priority elevation. As long as `epdc_panic` is asserted, transaction requests from the EPDC will have their priority elevated at the SoC-level bus arbitration control points.

The registers and associated functionality are as follows:

- EPDC\_FIFOCTRL[ENABLE\_PRIORITY] must be set to enable `epdc_panic` functionality.
- EPDC\_FIFOCTRL[FIFO\_L\_LEVEL] determines FIFO threshold which causes `epdc_panic` to assert when crossed.
- EPDC\_FIFOCTRL[FIFO\_H\_LEVEL] once `epdc_panic` has been asserted due to the pixel FIFO(s) crossing FIFO\_L\_LEVEL, this value, which must be set higher than FIFO\_L\_LEVEL, determines the point at which `epdc_panic` will de-assert as the FIFO(s) fills.

Note that the delta between FIFO\_H\_LEVEL and FIFO\_L\_LEVEL is intended as a de-bounce mechanism. The purpose is to avoid a constant panic situation, which would cause thrashing of the `epdc_panic` signal.

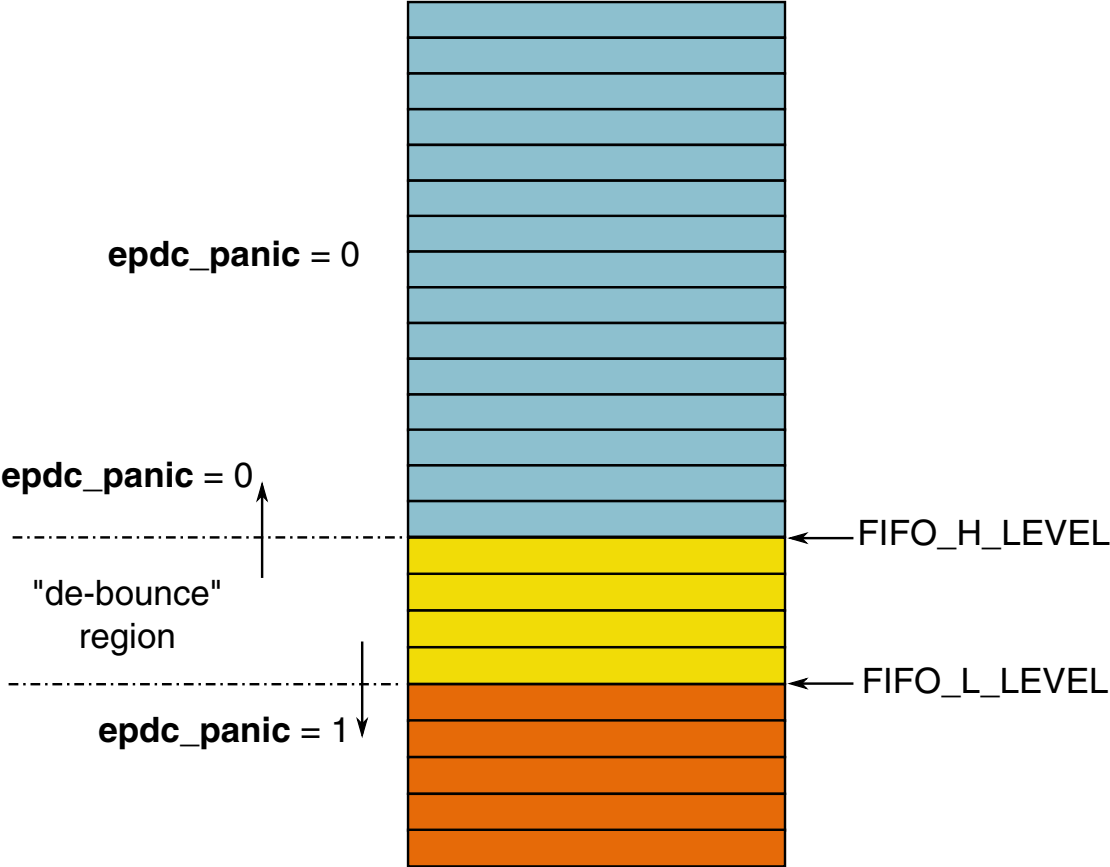


Figure 22-16. FIFO Priority Elevation

## 22.4 EPDC Memory Map/Register Definition

### EPDC Register Format Summary

EPDC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20F_4000	EPDC Control Register (EPDC_CTRL)	32	R/W	C000_0000h	<a href="#">22.4.1/951</a>
20F_4004	EPDC Control Register (EPDC_CTRL_SET)	32	R/W	C000_0000h	<a href="#">22.4.1/951</a>
20F_4008	EPDC Control Register (EPDC_CTRL_CLR)	32	R/W	C000_0000h	<a href="#">22.4.1/951</a>
20F_400C	EPDC Control Register (EPDC_CTRL_TOG)	32	R/W	C000_0000h	<a href="#">22.4.1/951</a>
20F_4020	EPDC Waveform Address Pointer (EPDC_WVADDR)	32	R/W	0000_0000h	<a href="#">22.4.2/952</a>
20F_4030	EPDC Working Buffer Address (EPDC_WB_ADDR)	32	R/W	0000_0000h	<a href="#">22.4.3/953</a>
20F_4040	EPDC Screen Resolution (EPDC_RES)	32	R/W	0000_0000h	<a href="#">22.4.4/954</a>

Table continues on the next page...

**EPDC memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/ page</b>
20F_4050	EPDC Format Control Register (EPDC_FORMAT)	32	R/W	0000_0000h	<a href="#">22.4.5/954</a>
20F_4054	EPDC Format Control Register (EPDC_FORMAT_SET)	32	R/W	0000_0000h	<a href="#">22.4.5/954</a>
20F_4058	EPDC Format Control Register (EPDC_FORMAT_CLR)	32	R/W	0000_0000h	<a href="#">22.4.5/954</a>
20F_405C	EPDC Format Control Register (EPDC_FORMAT_TOG)	32	R/W	0000_0000h	<a href="#">22.4.5/954</a>
20F_40A0	EPDC FIFO control register (EPDC_FIFOCtrl)	32	R/W	0080_0000h	<a href="#">22.4.6/956</a>
20F_40A4	EPDC FIFO control register (EPDC_FIFOCtrl_SET)	32	R/W	0080_0000h	<a href="#">22.4.6/956</a>
20F_40A8	EPDC FIFO control register (EPDC_FIFOCtrl_CLR)	32	R/W	0080_0000h	<a href="#">22.4.6/956</a>
20F_40AC	EPDC FIFO control register (EPDC_FIFOCtrl_TOG)	32	R/W	0080_0000h	<a href="#">22.4.6/956</a>
20F_4100	EPDC Update Region Address (EPDC_UPD_ADDR)	32	R/W	0000_0000h	<a href="#">22.4.7/957</a>
20F_4110	EPDC Update Region Stride (EPDC_UPD_STRIDE)	32	R/W	0000_0000h	<a href="#">22.4.8/958</a>
20F_4120	EPDC Update Command Co-ordinate (EPDC_UPD_CORD)	32	R/W	0000_0000h	<a href="#">22.4.9/958</a>
20F_4140	EPDC Update Command Size (EPDC_UPD_SIZE)	32	R/W	0000_0000h	<a href="#">22.4.10/959</a>
20F_4160	EPDC Update Command Control (EPDC_UPD_CTRL)	32	R/W	0000_0000h	<a href="#">22.4.11/960</a>
20F_4164	EPDC Update Command Control (EPDC_UPD_CTRL_SET)	32	R/W	0000_0000h	<a href="#">22.4.11/960</a>
20F_4168	EPDC Update Command Control (EPDC_UPD_CTRL_CLR)	32	R/W	0000_0000h	<a href="#">22.4.11/960</a>
20F_416C	EPDC Update Command Control (EPDC_UPD_CTRL_TOG)	32	R/W	0000_0000h	<a href="#">22.4.11/960</a>
20F_4180	EPDC Update Fixed Pixel Control (EPDC_UPD_FIXED)	32	R/W	0000_0000h	<a href="#">22.4.12/961</a>
20F_4184	EPDC Update Fixed Pixel Control (EPDC_UPD_FIXED_SET)	32	R/W	0000_0000h	<a href="#">22.4.12/961</a>
20F_4188	EPDC Update Fixed Pixel Control (EPDC_UPD_FIXED_CLR)	32	R/W	0000_0000h	<a href="#">22.4.12/961</a>
20F_418C	EPDC Update Fixed Pixel Control (EPDC_UPD_FIXED_TOG)	32	R/W	0000_0000h	<a href="#">22.4.12/961</a>
20F_41A0	EPDC Temperature Register (EPDC_TEMP)	32	R/W	0000_0000h	<a href="#">22.4.13/962</a>
20F_41C0	Waveform Mode Lookup Table Control Register. (EPDC_AUTOWV_LUT)	32	R/W	0000_0000h	<a href="#">22.4.14/962</a>
20F_4200	EPDC Timing Control Engine Control Register (EPDC_TCE_CTRL)	32	R/W	0000_0010h	<a href="#">22.4.15/963</a>
20F_4204	EPDC Timing Control Engine Control Register (EPDC_TCE_CTRL_SET)	32	R/W	0000_0010h	<a href="#">22.4.15/963</a>
20F_4208	EPDC Timing Control Engine Control Register (EPDC_TCE_CTRL_CLR)	32	R/W	0000_0010h	<a href="#">22.4.15/963</a>
20F_420C	EPDC Timing Control Engine Control Register (EPDC_TCE_CTRL_TOG)	32	R/W	0000_0010h	<a href="#">22.4.15/963</a>
20F_4220	EPDC Timing Control Engine Source-Driver Config Register (EPDC_TCE_SDCFG)	32	R/W	0000_0000h	<a href="#">22.4.16/965</a>

*Table continues on the next page...*

## EPDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20F_4224	EPDC Timing Control Engine Source-Driver Config Register (EPDC_TCE_SDCFG_SET)	32	R/W	0000_0000h	<a href="#">22.4.16/965</a>
20F_4228	EPDC Timing Control Engine Source-Driver Config Register (EPDC_TCE_SDCFG_CLR)	32	R/W	0000_0000h	<a href="#">22.4.16/965</a>
20F_422C	EPDC Timing Control Engine Source-Driver Config Register (EPDC_TCE_SDCFG_TOG)	32	R/W	0000_0000h	<a href="#">22.4.16/965</a>
20F_4240	EPDC Timing Control Engine Gate-Driver Config Register (EPDC_TCE_GDCFG)	32	R/W	0000_0000h	<a href="#">22.4.17/966</a>
20F_4244	EPDC Timing Control Engine Gate-Driver Config Register (EPDC_TCE_GDCFG_SET)	32	R/W	0000_0000h	<a href="#">22.4.17/966</a>
20F_4248	EPDC Timing Control Engine Gate-Driver Config Register (EPDC_TCE_GDCFG_CLR)	32	R/W	0000_0000h	<a href="#">22.4.17/966</a>
20F_424C	EPDC Timing Control Engine Gate-Driver Config Register (EPDC_TCE_GDCFG_TOG)	32	R/W	0000_0000h	<a href="#">22.4.17/966</a>
20F_4260	EPDC Timing Control Engine Horizontal Timing Register 1 (EPDC_TCE_HSCAN1)	32	R/W	0000_0000h	<a href="#">22.4.18/967</a>
20F_4264	EPDC Timing Control Engine Horizontal Timing Register 1 (EPDC_TCE_HSCAN1_SET)	32	R/W	0000_0000h	<a href="#">22.4.18/967</a>
20F_4268	EPDC Timing Control Engine Horizontal Timing Register 1 (EPDC_TCE_HSCAN1_CLR)	32	R/W	0000_0000h	<a href="#">22.4.18/967</a>
20F_426C	EPDC Timing Control Engine Horizontal Timing Register 1 (EPDC_TCE_HSCAN1_TOG)	32	R/W	0000_0000h	<a href="#">22.4.18/967</a>
20F_4280	EPDC Timing Control Engine Horizontal Timing Register 2 (EPDC_TCE_HSCAN2)	32	R/W	0000_0000h	<a href="#">22.4.19/967</a>
20F_4284	EPDC Timing Control Engine Horizontal Timing Register 2 (EPDC_TCE_HSCAN2_SET)	32	R/W	0000_0000h	<a href="#">22.4.19/967</a>
20F_4288	EPDC Timing Control Engine Horizontal Timing Register 2 (EPDC_TCE_HSCAN2_CLR)	32	R/W	0000_0000h	<a href="#">22.4.19/967</a>
20F_428C	EPDC Timing Control Engine Horizontal Timing Register 2 (EPDC_TCE_HSCAN2_TOG)	32	R/W	0000_0000h	<a href="#">22.4.19/967</a>
20F_42A0	EPDC Timing Control Engine Vertical Timing Register (EPDC_TCE_VSCAN)	32	R/W	0000_0000h	<a href="#">22.4.20/968</a>
20F_42A4	EPDC Timing Control Engine Vertical Timing Register (EPDC_TCE_VSCAN_SET)	32	R/W	0000_0000h	<a href="#">22.4.20/968</a>
20F_42A8	EPDC Timing Control Engine Vertical Timing Register (EPDC_TCE_VSCAN_CLR)	32	R/W	0000_0000h	<a href="#">22.4.20/968</a>
20F_42AC	EPDC Timing Control Engine Vertical Timing Register (EPDC_TCE_VSCAN_TOG)	32	R/W	0000_0000h	<a href="#">22.4.20/968</a>
20F_42C0	EPDC Timing Control Engine OE timing control Register (EPDC_TCE_OE)	32	R/W	0000_0000h	<a href="#">22.4.21/968</a>
20F_42C4	EPDC Timing Control Engine OE timing control Register (EPDC_TCE_OE_SET)	32	R/W	0000_0000h	<a href="#">22.4.21/968</a>
20F_42C8	EPDC Timing Control Engine OE timing control Register (EPDC_TCE_OE_CLR)	32	R/W	0000_0000h	<a href="#">22.4.21/968</a>

Table continues on the next page...

**EPDC memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/page</b>
20F_42CC	EPDC Timing Control Engine OE timing control Register (EPDC_TCE_OE_TOG)	32	R/W	0000_0000h	<a href="#">22.4.21/968</a>
20F_42E0	EPDC Timing Control Engine Driver Polarity Register (EPDC_TCE_POLARITY)	32	R/W	0000_001Eh	<a href="#">22.4.22/969</a>
20F_42E4	EPDC Timing Control Engine Driver Polarity Register (EPDC_TCE_POLARITY_SET)	32	R/W	0000_001Eh	<a href="#">22.4.22/969</a>
20F_42E8	EPDC Timing Control Engine Driver Polarity Register (EPDC_TCE_POLARITY_CLR)	32	R/W	0000_001Eh	<a href="#">22.4.22/969</a>
20F_42EC	EPDC Timing Control Engine Driver Polarity Register (EPDC_TCE_POLARITY_TOG)	32	R/W	0000_001Eh	<a href="#">22.4.22/969</a>
20F_4300	EPDC Timing Control Engine Timing Register 1 (EPDC_TCE_TIMING1)	32	R/W	0000_0000h	<a href="#">22.4.23/970</a>
20F_4304	EPDC Timing Control Engine Timing Register 1 (EPDC_TCE_TIMING1_SET)	32	R/W	0000_0000h	<a href="#">22.4.23/970</a>
20F_4308	EPDC Timing Control Engine Timing Register 1 (EPDC_TCE_TIMING1_CLR)	32	R/W	0000_0000h	<a href="#">22.4.23/970</a>
20F_430C	EPDC Timing Control Engine Timing Register 1 (EPDC_TCE_TIMING1_TOG)	32	R/W	0000_0000h	<a href="#">22.4.23/970</a>
20F_4310	EPDC Timing Control Engine Timing Register 2 (EPDC_TCE_TIMING2)	32	R/W	0000_0001h	<a href="#">22.4.24/971</a>
20F_4314	EPDC Timing Control Engine Timing Register 2 (EPDC_TCE_TIMING2_SET)	32	R/W	0000_0001h	<a href="#">22.4.24/971</a>
20F_4318	EPDC Timing Control Engine Timing Register 2 (EPDC_TCE_TIMING2_CLR)	32	R/W	0000_0001h	<a href="#">22.4.24/971</a>
20F_431C	EPDC Timing Control Engine Timing Register 2 (EPDC_TCE_TIMING2_TOG)	32	R/W	0000_0001h	<a href="#">22.4.24/971</a>
20F_4320	EPDC Timing Control Engine Timing Register 3 (EPDC_TCE_TIMING3)	32	R/W	0000_0001h	<a href="#">22.4.25/972</a>
20F_4324	EPDC Timing Control Engine Timing Register 3 (EPDC_TCE_TIMING3_SET)	32	R/W	0000_0001h	<a href="#">22.4.25/972</a>
20F_4328	EPDC Timing Control Engine Timing Register 3 (EPDC_TCE_TIMING3_CLR)	32	R/W	0000_0001h	<a href="#">22.4.25/972</a>
20F_432C	EPDC Timing Control Engine Timing Register 3 (EPDC_TCE_TIMING3_TOG)	32	R/W	0000_0001h	<a href="#">22.4.25/972</a>
20F_4380	EPDC Pigeon Mode Control Register 0 (EPDC_PIGEON_CTRL0)	32	R/W	0000_0000h	<a href="#">22.4.26/972</a>
20F_4384	EPDC Pigeon Mode Control Register 0 (EPDC_PIGEON_CTRL0_SET)	32	R/W	0000_0000h	<a href="#">22.4.26/972</a>
20F_4388	EPDC Pigeon Mode Control Register 0 (EPDC_PIGEON_CTRL0_CLR)	32	R/W	0000_0000h	<a href="#">22.4.26/972</a>
20F_438C	EPDC Pigeon Mode Control Register 0 (EPDC_PIGEON_CTRL0_TOG)	32	R/W	0000_0000h	<a href="#">22.4.26/972</a>
20F_4390	EPDC Pigeon Mode Control Register 1 (EPDC_PIGEON_CTRL1)	32	R/W	0000_0000h	<a href="#">22.4.27/973</a>

*Table continues on the next page...*

## EPDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20F_4394	EPDC Pigeon Mode Control Register 1 (EPDC_PIGEON_CTRL1_SET)	32	R/W	0000_0000h	<a href="#">22.4.27/973</a>
20F_4398	EPDC Pigeon Mode Control Register 1 (EPDC_PIGEON_CTRL1_CLR)	32	R/W	0000_0000h	<a href="#">22.4.27/973</a>
20F_439C	EPDC Pigeon Mode Control Register 1 (EPDC_PIGEON_CTRL1_TOG)	32	R/W	0000_0000h	<a href="#">22.4.27/973</a>
20F_43C0	EPDC IRQ Mask Register for LUT 0~31 (EPDC_IRQ_MASK1)	32	R/W	0000_0000h	<a href="#">22.4.28/973</a>
20F_43C4	EPDC IRQ Mask Register for LUT 0~31 (EPDC_IRQ_MASK1_SET)	32	R/W	0000_0000h	<a href="#">22.4.28/973</a>
20F_43C8	EPDC IRQ Mask Register for LUT 0~31 (EPDC_IRQ_MASK1_CLR)	32	R/W	0000_0000h	<a href="#">22.4.28/973</a>
20F_43CC	EPDC IRQ Mask Register for LUT 0~31 (EPDC_IRQ_MASK1_TOG)	32	R/W	0000_0000h	<a href="#">22.4.28/973</a>
20F_43D0	EPDC IRQ Mask Register for LUT 32~63 (EPDC_IRQ_MASK2)	32	R/W	0000_0000h	<a href="#">22.4.29/974</a>
20F_43D4	EPDC IRQ Mask Register for LUT 32~63 (EPDC_IRQ_MASK2_SET)	32	R/W	0000_0000h	<a href="#">22.4.29/974</a>
20F_43D8	EPDC IRQ Mask Register for LUT 32~63 (EPDC_IRQ_MASK2_CLR)	32	R/W	0000_0000h	<a href="#">22.4.29/974</a>
20F_43DC	EPDC IRQ Mask Register for LUT 32~63 (EPDC_IRQ_MASK2_TOG)	32	R/W	0000_0000h	<a href="#">22.4.29/974</a>
20F_43E0	EPDC Interrupt Register for LUT 0~31 (EPDC_IRQ1)	32	R/W	0000_0000h	<a href="#">22.4.30/974</a>
20F_43E4	EPDC Interrupt Register for LUT 0~31 (EPDC_IRQ1_SET)	32	R/W	0000_0000h	<a href="#">22.4.30/974</a>
20F_43E8	EPDC Interrupt Register for LUT 0~31 (EPDC_IRQ1_CLR)	32	R/W	0000_0000h	<a href="#">22.4.30/974</a>
20F_43EC	EPDC Interrupt Register for LUT 0~31 (EPDC_IRQ1_TOG)	32	R/W	0000_0000h	<a href="#">22.4.30/974</a>
20F_43F0	EPDC Interrupt Register for LUT 32~63 (EPDC_IRQ2)	32	R/W	0000_0000h	<a href="#">22.4.31/975</a>
20F_43F4	EPDC Interrupt Register for LUT 32~63 (EPDC_IRQ2_SET)	32	R/W	0000_0000h	<a href="#">22.4.31/975</a>
20F_43F8	EPDC Interrupt Register for LUT 32~63 (EPDC_IRQ2_CLR)	32	R/W	0000_0000h	<a href="#">22.4.31/975</a>
20F_43FC	EPDC Interrupt Register for LUT 32~63 (EPDC_IRQ2_TOG)	32	R/W	0000_0000h	<a href="#">22.4.31/975</a>
20F_4400	EPDC IRQ Mask Register (EPDC_IRQ_MASK)	32	R/W	0000_0000h	<a href="#">22.4.32/975</a>
20F_4404	EPDC IRQ Mask Register (EPDC_IRQ_MASK_SET)	32	R/W	0000_0000h	<a href="#">22.4.32/975</a>
20F_4408	EPDC IRQ Mask Register (EPDC_IRQ_MASK_CLR)	32	R/W	0000_0000h	<a href="#">22.4.32/975</a>

Table continues on the next page...

**EPDC memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/ page</b>
20F_440C	EPDC IRQ Mask Register (EPDC_IRQ_MASK_TOG)	32	R/W	0000_0000h	<a href="#">22.4.32/ 975</a>
20F_4420	EPDC Interrupt Register (EPDC_IRQ)	32	R/W	0000_0000h	<a href="#">22.4.33/ 976</a>
20F_4424	EPDC Interrupt Register (EPDC_IRQ_SET)	32	R/W	0000_0000h	<a href="#">22.4.33/ 976</a>
20F_4428	EPDC Interrupt Register (EPDC_IRQ_CLR)	32	R/W	0000_0000h	<a href="#">22.4.33/ 976</a>
20F_442C	EPDC Interrupt Register (EPDC_IRQ_TOG)	32	R/W	0000_0000h	<a href="#">22.4.33/ 976</a>
20F_4440	EPDC Status Register - LUTs (EPDC_STATUS_LUTS1)	32	R/W	0000_0000h	<a href="#">22.4.34/ 977</a>
20F_4444	EPDC Status Register - LUTs (EPDC_STATUS_LUTS1_SET)	32	R/W	0000_0000h	<a href="#">22.4.34/ 977</a>
20F_4448	EPDC Status Register - LUTs (EPDC_STATUS_LUTS1_CLR)	32	R/W	0000_0000h	<a href="#">22.4.34/ 977</a>
20F_444C	EPDC Status Register - LUTs (EPDC_STATUS_LUTS1_TOG)	32	R/W	0000_0000h	<a href="#">22.4.34/ 977</a>
20F_4450	EPDC Status Register - LUTs (EPDC_STATUS_LUTS2)	32	R/W	0000_0000h	<a href="#">22.4.35/ 978</a>
20F_4454	EPDC Status Register - LUTs (EPDC_STATUS_LUTS2_SET)	32	R/W	0000_0000h	<a href="#">22.4.35/ 978</a>
20F_4458	EPDC Status Register - LUTs (EPDC_STATUS_LUTS2_CLR)	32	R/W	0000_0000h	<a href="#">22.4.35/ 978</a>
20F_445C	EPDC Status Register - LUTs (EPDC_STATUS_LUTS2_TOG)	32	R/W	0000_0000h	<a href="#">22.4.35/ 978</a>
20F_4460	EPDC Status Register - Next Available LUT (EPDC_STATUS_NEXTLUT)	32	R/W	0000_013Fh	<a href="#">22.4.36/ 978</a>
20F_4480	EPDC LUT Collision Status (EPDC_STATUS_COL1)	32	R/W	0000_0000h	<a href="#">22.4.37/ 980</a>
20F_4484	EPDC LUT Collision Status (EPDC_STATUS_COL1_SET)	32	R/W	0000_0000h	<a href="#">22.4.37/ 980</a>
20F_4488	EPDC LUT Collision Status (EPDC_STATUS_COL1_CLR)	32	R/W	0000_0000h	<a href="#">22.4.37/ 980</a>
20F_448C	EPDC LUT Collision Status (EPDC_STATUS_COL1_TOG)	32	R/W	0000_0000h	<a href="#">22.4.37/ 980</a>
20F_4490	EPDC LUT Collision Status (EPDC_STATUS_COL2)	32	R/W	0000_0000h	<a href="#">22.4.38/ 981</a>
20F_4494	EPDC LUT Collision Status (EPDC_STATUS_COL2_SET)	32	R/W	0000_0000h	<a href="#">22.4.38/ 981</a>
20F_4498	EPDC LUT Collision Status (EPDC_STATUS_COL2_CLR)	32	R/W	0000_0000h	<a href="#">22.4.38/ 981</a>
20F_449C	EPDC LUT Collision Status (EPDC_STATUS_COL2_TOG)	32	R/W	0000_0000h	<a href="#">22.4.38/ 981</a>

*Table continues on the next page...*

## EPDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20F_44A0	EPDC General Status Register (EPDC_STATUS)	32	R/W	0000_0008h	<a href="#">22.4.39/981</a>
20F_44A4	EPDC General Status Register (EPDC_STATUS_SET)	32	R/W	0000_0008h	<a href="#">22.4.39/981</a>
20F_44A8	EPDC General Status Register (EPDC_STATUS_CLR)	32	R/W	0000_0008h	<a href="#">22.4.39/981</a>
20F_44AC	EPDC General Status Register (EPDC_STATUS_TOG)	32	R/W	0000_0008h	<a href="#">22.4.39/981</a>
20F_44C0	EPDC Collision Region Co-ordinate (EPDC_UPD_COL_CORD)	32	R/W	0000_0000h	<a href="#">22.4.40/983</a>
20F_44E0	EPDC Collision Region Size (EPDC_UPD_COL_SIZE)	32	R/W	0000_0000h	<a href="#">22.4.41/984</a>
20F_4600	1-level Histogram Parameter Register. (EPDC_HIST1_PARAM)	32	R/W	0000_0000h	<a href="#">22.4.42/984</a>
20F_4610	2-level Histogram Parameter Register. (EPDC_HIST2_PARAM)	32	R/W	0000_0F00h	<a href="#">22.4.43/985</a>
20F_4620	4-level Histogram Parameter Register. (EPDC_HIST4_PARAM)	32	R/W	0F0A_0500h	<a href="#">22.4.44/986</a>
20F_4630	8-level Histogram Parameter 0 Register. (EPDC_HIST8_PARAM0)	32	R/W	0604_0200h	<a href="#">22.4.45/986</a>
20F_4640	8-level Histogram Parameter 1 Register. (EPDC_HIST8_PARAM1)	32	R/W	0F0D_0B09h	<a href="#">22.4.46/987</a>
20F_4650	16-level Histogram Parameter 0 Register. (EPDC_HIST16_PARAM0)	32	R/W	0302_0100h	<a href="#">22.4.47/988</a>
20F_4660	16-level Histogram Parameter Register. (EPDC_HIST16_PARAM1)	32	R/W	0706_0504h	<a href="#">22.4.48/989</a>
20F_4670	16-level Histogram Parameter Register. (EPDC_HIST16_PARAM2)	32	R/W	0B0A_0908h	<a href="#">22.4.49/990</a>
20F_4680	16-level Histogram Parameter Register. (EPDC_HIST16_PARAM3)	32	R/W	0F0E_0D0Ch	<a href="#">22.4.50/990</a>
20F_4700	EPDC General Purpose I/O Debug register (EPDC_GPIO)	32	R/W	0000_0000h	<a href="#">22.4.51/991</a>
20F_4704	EPDC General Purpose I/O Debug register (EPDC_GPIO_SET)	32	R/W	0000_0000h	<a href="#">22.4.51/991</a>
20F_4708	EPDC General Purpose I/O Debug register (EPDC_GPIO_CLR)	32	R/W	0000_0000h	<a href="#">22.4.51/991</a>
20F_470C	EPDC General Purpose I/O Debug register (EPDC_GPIO_TOG)	32	R/W	0000_0000h	<a href="#">22.4.51/991</a>
20F_47F0	EPDC Version Register (EPDC_VERSION)	32	R/W	0200_0000h	<a href="#">22.4.52/992</a>
20F_4800	Panel Interface Signal Generator Register 0_0 (EPDC_PIGEON_0_0)	32	R/W	0000_0F00h	<a href="#">22.4.53/993</a>
20F_4810	Panel Interface Signal Generator Register 0_1 (EPDC_PIGEON_0_1)	32	R/W	0000_0000h	<a href="#">22.4.54/994</a>

Table continues on the next page...



## EPDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20F_4820	Panel Interface Signal Generator Register 0_1 (EPDC_PIGEON_0_2)	32	R/W	0000_0000h	<a href="#">22.4.55/995</a>
20F_4840	Panel Interface Signal Generator Register 1_0 (EPDC_PIGEON_1_0)	32	R/W	0000_0F00h	<a href="#">22.4.56/995</a>
20F_4850	Panel Interface Signal Generator Register 1_1 (EPDC_PIGEON_1_1)	32	R/W	0000_0000h	<a href="#">22.4.57/996</a>
20F_4860	Panel Interface Signal Generator Register 1_1 (EPDC_PIGEON_1_2)	32	R/W	0000_0000h	<a href="#">22.4.58/997</a>
20F_4880	Panel Interface Signal Generator Register 2_0 (EPDC_PIGEON_2_0)	32	R/W	0000_0F00h	<a href="#">22.4.59/998</a>
20F_4890	Panel Interface Signal Generator Register 2_1 (EPDC_PIGEON_2_1)	32	R/W	0000_0000h	<a href="#">22.4.60/999</a>
20F_48A0	Panel Interface Signal Generator Register 2_1 (EPDC_PIGEON_2_2)	32	R/W	0000_0000h	<a href="#">22.4.61/999</a>
20F_48C0	Panel Interface Signal Generator Register 3_0 (EPDC_PIGEON_3_0)	32	R/W	0000_0F00h	<a href="#">22.4.62/1000</a>
20F_48D0	Panel Interface Signal Generator Register 3_1 (EPDC_PIGEON_3_1)	32	R/W	0000_0000h	<a href="#">22.4.63/1001</a>
20F_48E0	Panel Interface Signal Generator Register 3_1 (EPDC_PIGEON_3_2)	32	R/W	0000_0000h	<a href="#">22.4.64/1002</a>
20F_4900	Panel Interface Signal Generator Register 4_0 (EPDC_PIGEON_4_0)	32	R/W	0000_0F00h	<a href="#">22.4.65/1002</a>
20F_4910	Panel Interface Signal Generator Register 4_1 (EPDC_PIGEON_4_1)	32	R/W	0000_0000h	<a href="#">22.4.66/1004</a>
20F_4920	Panel Interface Signal Generator Register 4_1 (EPDC_PIGEON_4_2)	32	R/W	0000_0000h	<a href="#">22.4.67/1004</a>
20F_4940	Panel Interface Signal Generator Register 5_0 (EPDC_PIGEON_5_0)	32	R/W	0000_0F00h	<a href="#">22.4.68/1005</a>
20F_4950	Panel Interface Signal Generator Register 5_1 (EPDC_PIGEON_5_1)	32	R/W	0000_0000h	<a href="#">22.4.69/1006</a>
20F_4960	Panel Interface Signal Generator Register 5_1 (EPDC_PIGEON_5_2)	32	R/W	0000_0000h	<a href="#">22.4.70/1007</a>
20F_4980	Panel Interface Signal Generator Register 6_0 (EPDC_PIGEON_6_0)	32	R/W	0000_0F00h	<a href="#">22.4.71/1007</a>
20F_4990	Panel Interface Signal Generator Register 6_1 (EPDC_PIGEON_6_1)	32	R/W	0000_0000h	<a href="#">22.4.72/1008</a>
20F_49A0	Panel Interface Signal Generator Register 6_1 (EPDC_PIGEON_6_2)	32	R/W	0000_0000h	<a href="#">22.4.73/1009</a>
20F_49C0	Panel Interface Signal Generator Register 7_0 (EPDC_PIGEON_7_0)	32	R/W	0000_0F00h	<a href="#">22.4.74/1010</a>
20F_49D0	Panel Interface Signal Generator Register 7_1 (EPDC_PIGEON_7_1)	32	R/W	0000_0000h	<a href="#">22.4.75/1011</a>
20F_49E0	Panel Interface Signal Generator Register 7_1 (EPDC_PIGEON_7_2)	32	R/W	0000_0000h	<a href="#">22.4.76/1011</a>

Table continues on the next page...

## EPDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20F_4A00	Panel Interface Signal Generator Register 8_0 (EPDC_PIGEON_8_0)	32	R/W	0000_0F00h	<a href="#">22.4.77/1012</a>
20F_4A10	Panel Interface Signal Generator Register 8_1 (EPDC_PIGEON_8_1)	32	R/W	0000_0000h	<a href="#">22.4.78/1013</a>
20F_4A20	Panel Interface Signal Generator Register 8_1 (EPDC_PIGEON_8_2)	32	R/W	0000_0000h	<a href="#">22.4.79/1014</a>
20F_4A40	Panel Interface Signal Generator Register 9_0 (EPDC_PIGEON_9_0)	32	R/W	0000_0F00h	<a href="#">22.4.80/1014</a>
20F_4A50	Panel Interface Signal Generator Register 9_1 (EPDC_PIGEON_9_1)	32	R/W	0000_0000h	<a href="#">22.4.81/1016</a>
20F_4A60	Panel Interface Signal Generator Register 9_1 (EPDC_PIGEON_9_2)	32	R/W	0000_0000h	<a href="#">22.4.82/1016</a>
20F_4A80	Panel Interface Signal Generator Register 10_0 (EPDC_PIGEON_10_0)	32	R/W	0000_0F00h	<a href="#">22.4.83/1017</a>
20F_4A90	Panel Interface Signal Generator Register 10_1 (EPDC_PIGEON_10_1)	32	R/W	0000_0000h	<a href="#">22.4.84/1018</a>
20F_4AA0	Panel Interface Signal Generator Register 10_1 (EPDC_PIGEON_10_2)	32	R/W	0000_0000h	<a href="#">22.4.85/1019</a>
20F_4AC0	Panel Interface Signal Generator Register 11_0 (EPDC_PIGEON_11_0)	32	R/W	0000_0F00h	<a href="#">22.4.86/1019</a>
20F_4AD0	Panel Interface Signal Generator Register 11_1 (EPDC_PIGEON_11_1)	32	R/W	0000_0000h	<a href="#">22.4.87/1020</a>
20F_4AE0	Panel Interface Signal Generator Register 11_1 (EPDC_PIGEON_11_2)	32	R/W	0000_0000h	<a href="#">22.4.88/1021</a>
20F_4B00	Panel Interface Signal Generator Register 12_0 (EPDC_PIGEON_12_0)	32	R/W	0000_0F00h	<a href="#">22.4.89/1022</a>
20F_4B10	Panel Interface Signal Generator Register 12_1 (EPDC_PIGEON_12_1)	32	R/W	0000_0000h	<a href="#">22.4.90/1023</a>
20F_4B20	Panel Interface Signal Generator Register 12_1 (EPDC_PIGEON_12_2)	32	R/W	0000_0000h	<a href="#">22.4.91/1023</a>
20F_4B40	Panel Interface Signal Generator Register 13_0 (EPDC_PIGEON_13_0)	32	R/W	0000_0F00h	<a href="#">22.4.92/1024</a>
20F_4B50	Panel Interface Signal Generator Register 13_1 (EPDC_PIGEON_13_1)	32	R/W	0000_0000h	<a href="#">22.4.93/1025</a>
20F_4B60	Panel Interface Signal Generator Register 13_1 (EPDC_PIGEON_13_2)	32	R/W	0000_0000h	<a href="#">22.4.94/1026</a>
20F_4B80	Panel Interface Signal Generator Register 14_0 (EPDC_PIGEON_14_0)	32	R/W	0000_0F00h	<a href="#">22.4.95/1026</a>
20F_4B90	Panel Interface Signal Generator Register 14_1 (EPDC_PIGEON_14_1)	32	R/W	0000_0000h	<a href="#">22.4.96/1028</a>
20F_4BA0	Panel Interface Signal Generator Register 14_1 (EPDC_PIGEON_14_2)	32	R/W	0000_0000h	<a href="#">22.4.97/1028</a>
20F_4BC0	Panel Interface Signal Generator Register 15_0 (EPDC_PIGEON_15_0)	32	R/W	0000_0F00h	<a href="#">22.4.98/1029</a>

Table continues on the next page...

**EPDC memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20F_4BD0	Panel Interface Signal Generator Register 15_1 (EPDC_PIGEON_15_1)	32	R/W	0000_0000h	<a href="#">22.4.99/1030</a>
20F_4BE0	Panel Interface Signal Generator Register 15_2 (EPDC_PIGEON_15_2)	32	R/W	0000_0000h	<a href="#">22.4.100/1031</a>
20F_4C10	EPDC Working Buffer Address for TCE (EPDC_WB_ADDR_TCE)	32	R/W	0000_0000h	<a href="#">22.4.101/1031</a>

**22.4.1 EPDC Control Register (EPDC\_CTRL<sub>n</sub>)**

EPDC Main control register

This register controls various high-level functions of the EPDC

**EXAMPLE**

```

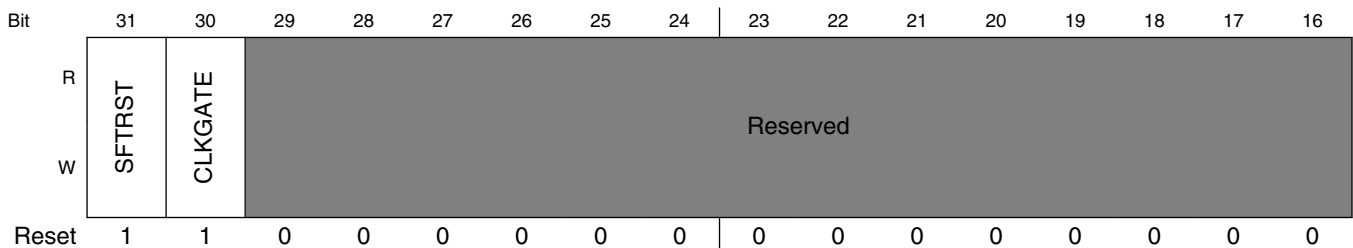
/* Reset */
__raw_writel(EPDC_CTRL_SFTRST, EPDC_CTRL_SET);
while (!(__raw_readl(EPDC_CTRL) & EPDC_CTRL_CLKGATE))
;
__raw_writel(EPDC_CTRL_SFTRST, EPDC_CTRL_CLEAR);

/* Enable clock gating (clear to enable) */
__raw_writel(EPDC_CTRL_CLKGATE, EPDC_CTRL_CLEAR);
while (__raw_readl(EPDC_CTRL) & (EPDC_CTRL_SFTRST | EPDC_CTRL_CLKGATE))
;

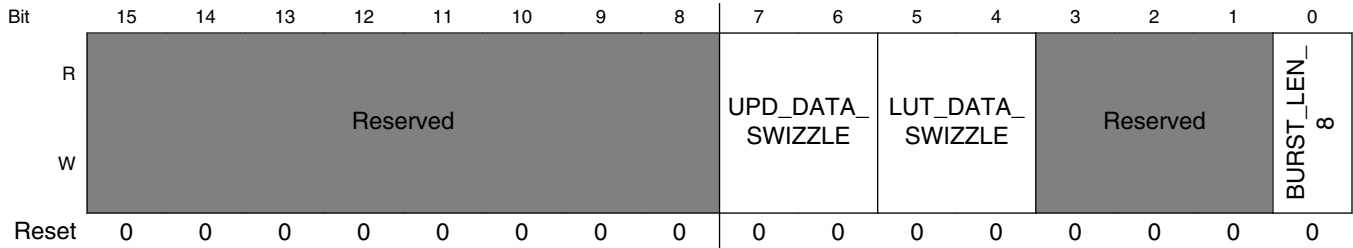
/* EPDC_CTRL */
reg_val = __raw_readl(EPDC_CTRL);
reg_val &= ~EPDC_CTRL_UPD_DATA_SWIZZLE_MASK;
reg_val |= EPDC_CTRL_UPD_DATA_SWIZZLE_NO_SWAP;
reg_val &= ~EPDC_CTRL_LUT_DATA_SWIZZLE_MASK;
reg_val |= EPDC_CTRL_LUT_DATA_SWIZZLE_NO_SWAP;
__raw_writel(reg_val, EPDC_CTRL_SET);

```

Address: 20F\_4000h base + 0h offset + (4d × i), where i=0d to 3d



## EPDC Memory Map/Register Definition



### EPDC\_CTRLn field descriptions

Field	Description
31 SFTRST	Set this bit to zero to enable normal EPDC operation. Set this bit to one (default) to disable clocking with the EPDC and hold it in its reset (lowest power) state. This bit can be turned on and then off to reset the EPDC block to its default state.
30 CLKGATE	This bit must be set to zero for normal operation. When set to one it gates off the clocks to the block.
29–8 -	This field is reserved. Reserved
7–6 UPD_DATA_SWIZZLE	Specifies how to swap the bytes for the UPD data before the WB construction. Plesae note this swizzle operate right after data fetch from bus, no matter it's aligned access or not. Supported configurations:  0x0 <b>NO_SWAP</b> — No byte swapping.(Little endian) 0x1 <b>ALL_BYTES_SWAP</b> — Swizzle all bytes, swap bytes 0,3 and 1,2 (aka Big Endian). 0x2 <b>HWD_SWAP</b> — Swap half-words. 0x3 <b>HWD_BYTE_SWAP</b> — Swap bytes within each half-word.
5–4 LUT_DATA_SWIZZLE	Specifies how to swap the bytes for the LUT data before store to LUTRAM. Supported configurations:  0x0 <b>NO_SWAP</b> — No byte swapping.(Little endian) 0x1 <b>ALL_BYTES_SWAP</b> — Swizzle all bytes, swap bytes 0,3 and 1,2 (aka Big Endian). 0x2 <b>HWD_SWAP</b> — Swap half-words. 0x3 <b>HWD_BYTE_SWAP</b> — Swap bytes within each half-word.
3–1 -	This field is reserved. Reserved.
0 BURST_LEN_8	0- EPDC display fifo logic will issue AXI bursts of length 16. When set to 1, the block will issue bursts of length 8.

## 22.4.2 EPDC Waveform Address Pointer (EPDC\_WVADDR)

### EPDC Waveform Address Pointer

#### EXAMPLE

```

/* Reset */
__raw_writel(EPDC_CTRL_SFTRST, EPDC_CTRL_SET);
while (!(__raw_readl(EPDC_CTRL) & EPDC_CTRL_CLKGATE))
;
__raw_writel(EPDC_CTRL_SFTRST, EPDC_CTRL_CLEAR);

/* Enable clock gating (clear to enable) */
__raw_writel(EPDC_CTRL_CLKGATE, EPDC_CTRL_CLEAR);

```

```

while (__raw_readl(EPDC_CTRL) & (EPDC_CTRL_SFTRST | EPDC_CTRL_CLKGATE))
;

/* EPDC_CTRL */
reg_val = __raw_readl(EPDC_CTRL);
reg_val &= ~EPDC_CTRL_UPD_DATA_SWIZZLE_MASK;
reg_val |= EPDC_CTRL_UPD_DATA_SWIZZLE_NO_SWAP;
reg_val &= ~EPDC_CTRL_LUT_DATA_SWIZZLE_MASK;
reg_val |= EPDC_CTRL_LUT_DATA_SWIZZLE_NO_SWAP;
__raw_writel(reg_val, EPDC_CTRL_SET);

```

Address: 20F\_4000h base + 20h offset = 20F\_4020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_WVADDR field descriptions

Field	Description
31–0 ADDR	Start address of waveform tables. This address needs to be aligned to a 64-bit word boundary.

## 22.4.3 EPDC Working Buffer Address (EPDC\_WB\_ADDR)

### EPDC Working Buffer Address

Address: 20F\_4000h base + 30h offset = 20F\_4030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

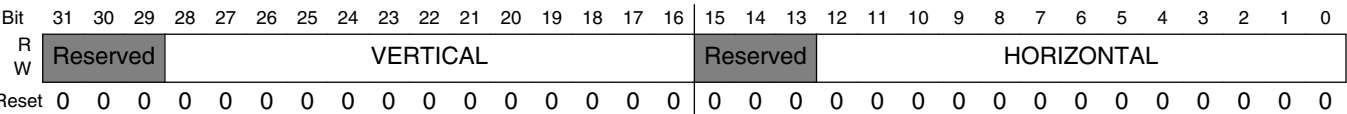
### EPDC\_WB\_ADDR field descriptions

Field	Description
31–0 ADDR	Address for EPDC working buffer. This address must be aligned to a 64-bit double-word boundary.

### 22.4.4 EPDC Screen Resolution (EPDC\_RES)

EPDC Screen Resolution. This register defines the horizontal and vertical resolution of the target display panel

Address: 20F\_4000h base + 40h offset = 20F\_4040h



EPDC\_RES field descriptions

Field	Description
31–29 -	This field is reserved. Reserved.
28–16 VERTICAL	Vertical Resolution (in pixels)
15–13 -	This field is reserved. Reserved.
12–0 HORIZONTAL	Horizontal Resolution (in pixels)

### 22.4.5 EPDC Format Control Register (EPDC\_FORMATn)

EPDC Pixel format control register. Defines formats for buffer and TFT pixels.

This register controls various functions throughout the digital portion of the chip.

EXAMPLE

```
/* EPDC_FORMAT - 2bit TFT and 4bit Buf pixel format */
reg_val = EPDC_FORMAT_TFT_PIXEL_FORMAT_2BIT
    | EPDC_FORMAT_BUF_PIXEL_FORMAT_P4N
    | ((0x0 << EPDC_FORMAT_DEFAULT_TFT_PIXEL_OFFSET) &
    EPDC_FORMAT_DEFAULT_TFT_PIXEL_MASK);
__raw_writel(reg_val, EPDC_FORMAT);
```

Address: 20F\_4000h base + 50h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								BUF_PIXEL_ SCALE	DEFAULT_TFT_PIXEL						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						BUF_PIXEL_ FORMAT		Reserved						TFT_PIXEL_ FORMAT	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EPDC\_FORMAT $n$  field descriptions**

Field	Description
31–25 -	This field is reserved. Reserved.
24 BUF_PIXEL_ SCALE	Selects method of conversion from 8-bit input  0x0 <b>TRUNCATE</b> — Use Truncate method (LSB) 0x1 <b>ROUND</b> — Use rounding method (with saturation)
23–16 DEFAULT_TFT_ PIXEL	Default TFT pixel value. This value is used as the source-driver voltage value (TFT-pixel) for either partial-updates where a pixel has not changed or for any part of the screen which is not being updated during active frame scans.
15–11 -	This field is reserved. Reserved.
10–8 BUF_PIXEL_ FORMAT	EPDC Input Buffer Pixel format. All update buffers are expected to have 8-bit grayscale pixels. This register defines which MSB's of those pixels are used. It must be noted that this format must match the waveform (e.g. P4N is not compatible with 3-bit waveforms)  0x2 <b>P2N</b> — 2-bit pixel 0x3 <b>P3N</b> — 3-bit pixel 0x4 <b>P4N</b> — 4-bit pixel 0x5 <b>P5N</b> — 5-bit pixel
7–2 -	This field is reserved. Reserved.
1–0 TFT_PIXEL_ FORMAT	EPDC TFT Pixel Format. This defines how many bits of the DATA bus are required per pixel. This field must be consistent with the waveform and panel architecture.  0x0 <b>2B</b> — 2-bit 0x1 <b>2BV</b> — 2-bit and VCOM 0x2 <b>4B</b> — 4-bit 0x3 <b>4BV</b> — 4-bit and VCOM

22.4.6 EPDC FIFO control register (EPDC\_FIFOCTRLn)

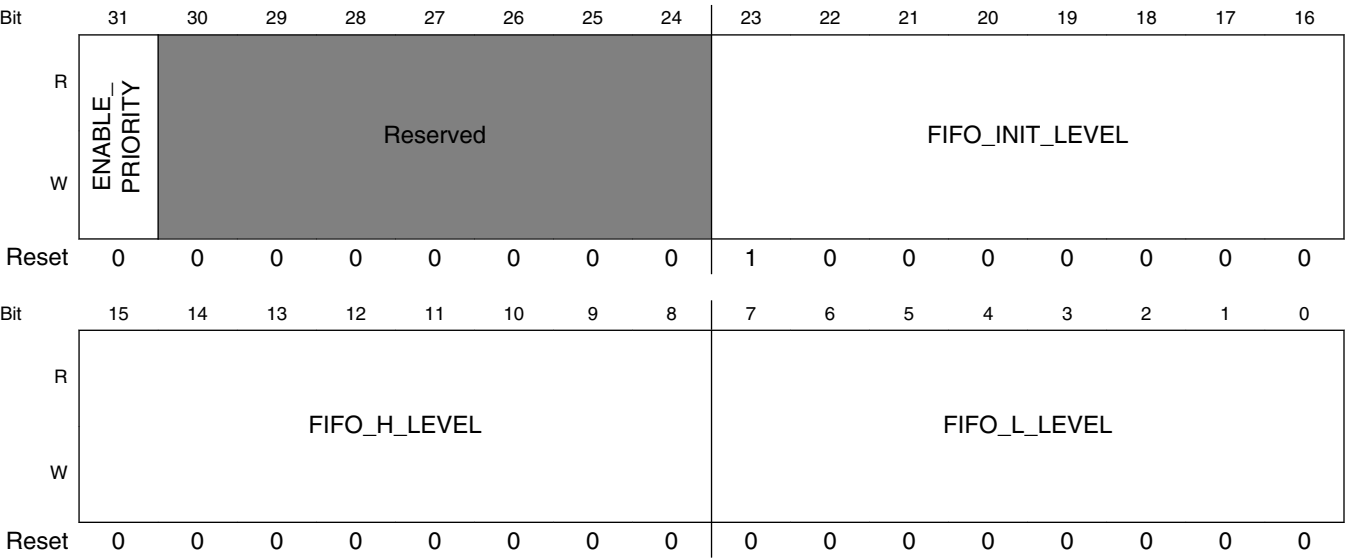
Allows for programmability of pixel FIFO watermarks used in conjunction with system arbitration hardware

This register houses FIFO control bits

EXAMPLE

```
/* EPDC_FIFOCTRL (disabled) */
reg_val =
  ((100 << EPDC_FIFOCTRL_FIFO_INIT_LEVEL_OFFSET) &
   EPDC_FIFOCTRL_FIFO_INIT_LEVEL_MASK)
  | ((200 << EPDC_FIFOCTRL_FIFO_H_LEVEL_OFFSET) &
   EPDC_FIFOCTRL_FIFO_H_LEVEL_MASK)
  | ((100 << EPDC_FIFOCTRL_FIFO_L_LEVEL_OFFSET) &
   EPDC_FIFOCTRL_FIFO_L_LEVEL_MASK);
__raw_writel(reg_val, EPDC_FIFOCTRL);
```

Address: 20F\_4000h base + A0h offset + (4d × i), where i=0d to 3d



EPDC\_FIFOCTRLn field descriptions

Field	Description
31 ENABLE_PRIORITY	Enable watermark-based priority elevation mechanism. 1=Enabled, 0=Disabled. (Only applies to FIFO_H_LEVEL and FIFO_L_LEVEL)
30–24 -	This field is reserved. Reserved.
23–16 FIFO_INIT_LEVEL	This register sets the watermark for the pixel-fifo.
15–8 FIFO_H_LEVEL	Upper level value of FIFO watermark. Must be greater than FIFO_L_LEVEL. When the pixel FIFO reaches this level or above, the priority elevation request is negated

Table continues on the next page...



**EPDC\_FIFOCTRLn field descriptions (continued)**

Field	Description
7–0 FIFO_L_LEVEL	Lower level value of FIFO watermark. When the pixel FIFO reaches this level or below, the priority elevation request is asserted.

**22.4.7 EPDC Update Region Address (EPDC\_UPD\_ADDR)****EPDC Update Region Address**

The address alignment depends on whether the update region stride register (EPDC\_UPD\_STRIDE) is set or not.

If EPDC\_UPD\_STRIDE is zero, the update region address must start and end on the 8-byte aligned addresses.

If EPDC\_UPD\_STRIDE is non-zero, the update region address can start and/or end on any byte addressing. But aligned address still recommended to get best bus performance.

**EXAMPLE1**

```
/* no update stride case */
u32 stride = 0;
u32 addr = 0x100; /* 8-byte aligned */
__raw_writel(stride, EPDC_UPD_STRIDE);
__raw_writel(addr, EPDC_UPD_ADDR);
```

**EXAMPLE2**

```
/* update stride = 345 */
u32 stride = 345;
u32 addr = 0x123; /* 8-byte unaligned is OK */
__raw_writel(stride, EPDC_UPD_STRIDE);
__raw_writel(addr, EPDC_UPD_ADDR);
```

Address: 20F\_4000h base + 100h offset = 20F\_4100h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	<div>ADDR</div>																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EPDC\_UPD\_ADDR field descriptions**

Field	Description
31–0 ADDR	Address for incoming region update. This address points to update region which will be processed into the working buffer.

## 22.4.8 EPDC Update Region Stride (EPDC\_UPD\_STRIDE)

### EPDC Update Region Stride

When UPD\_STRIDE==0 (stride feature disabled), UPD buffer line must start from 64-bit boundary and end on 64-bit boundary(padding if not). When UPD\_STRIDE!=0 (stride feature enabled), UPD buffer line can start or end on any byte address, UPD\_WIDTH should be set to real line bytes count as normal, while UPD\_STRIDE set to byte distance between two lines' start.

### EXAMPLE

see details on stride feature introduction

Address: 20F\_4000h base + 110h offset = 20F\_4110h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STRIDE																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### EPDC\_UPD\_STRIDE field descriptions

Field	Description
31–0 STRIDE	line stride for incoming region update

## 22.4.9 EPDC Update Command Co-ordinate (EPDC\_UPD\_CORD)

### EPDC Update Command Co-ordinate

This register is used for setting up the coordinate of a new update region.

Address: 20F\_4000h base + 120h offset = 20F\_4120h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				YCORD												Reserved				XCORD											
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_UPD\_CORD field descriptions

Field	Description
31–29 -	This field is reserved. Reserved.
28–16 YCORD	Y co-ordinate for incoming region update

Table continues on the next page...

**EPDC\_UPD\_CORD field descriptions (continued)**

Field	Description
15–13 -	This field is reserved. Reserved.
12–0 XCORD	X co-ordinate for incoming region update

**22.4.10 EPDC Update Command Size (EPDC\_UPD\_SIZE)**

## EPDC Update Command Size

This register sets up the size of an update region.

Address: 20F\_4000h base + 140h offset = 20F\_4140h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved			HEIGHT													Reserved			WIDTH												
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**EPDC\_UPD\_SIZE field descriptions**

Field	Description
31–29 -	This field is reserved. Reserved.
28–16 HEIGHT	Height (in pixels)
15–13 -	This field is reserved. Reserved.
12–0 WIDTH	Width (in pixels)

## 22.4.11 EPDC Update Command Control (EPDC\_UPD\_CTRLn)

EPDC Update Command Control. Writing to this registers triggers and update request operation.

Address: 20F\_4000h base + 160h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	USE_FIXED	Reserved										LUT_SEL				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WAVEFORM_MODE								Reserved			NO_LUT_CANCEL	AUTOWV_PAUSE	AUTOWV	DRY_RUN	UPDATE_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EPDC\_UPD\_CTRLn field descriptions**

Field	Description
31 USE_FIXED	Use fixed pixel values (requires programming of EPDC_UPD_FIXED)
30–22 -	This field is reserved. Reserved.
21–16 LUT_SEL	LUT select 0-63
15–8 WAVEFORM_MODE	Waveform Mode 0-255
7–5 -	This field is reserved. Reserved.
4 NO_LUT_CANCEL	EPDC will cancel LUT loading for void update (no real update needed because of partial or collision), set this bit to 1 to disable this feature
3 AUTOWV_PAUSE	0x0 <b>AUTO</b> — epdc will analyze update buffer, report histogram, then update waveform mode using the programmed mode mapping in AUTOWV_LUT and start LUT loading 0x1 <b>MANUAL</b> — epdc will analyze update buffer, report histogram, then pause and waiting software to write again with selected waveform mode to start lut loading

*Table continues on the next page...*

**EPDC\_UPD\_CTRL $n$  field descriptions (continued)**

Field	Description
2 AUTOWV	enable automatical waveform mode selection
1 DRY_RUN	Enable Dry Run mode(set to 1). WB won't be updated in this mode, and lut_sel will be ignored, so actually you don't need to wait for LUT available to use this feature
0 UPDATE_MODE	Update Mode 0x0 <b>PARTIAL</b> — Partial Update : only process changed pixels in region 0x1 <b>FULL</b> — Full Update : process all pixels in region

**22.4.12 EPDC Update Fixed Pixel Control (EPDC\_UPD\_FIXED $n$ )**

EPDC Update Control register for fixed-pixel updates (enabled via EPDC\_UPD\_CTRL[USE\_FIXED])

Address: 20F\_4000h base + 180h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	FIXNP_EN	FIXCP_EN	Reserved													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FIXNP								FIXCP							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EPDC\_UPD\_FIXED $n$  field descriptions**

Field	Description
31 FIXNP_EN	If set to 1, current updated region has the NP value defined by FIXNP
30 FIXCP_EN	If set to 1, current updated region has the CP value defined by FIXCP
29–16 -	This field is reserved. Reserved.
15–8 FIXNP	NP value if fixenp_en is set to 1. Data in Y8 format.
7–0 FIXCP	CP value if fixecp_en is set to 1. Data in Y8 format.

## 22.4.13 EPDC Temperature Register (EPDC\_TEMP)

### EPDC Temperature Compensation Register

Address: 20F\_4000h base + 1A0h offset = 20F\_41A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TEMPERATURE																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### EPDC\_TEMP field descriptions

Field	Description
31–0 TEMPERATURE	Temperature Value. This value is simply an index (not a temperature value). The index is used by the EPDC to access the correct temperature compensated waveform.

## 22.4.14 Waveform Mode Lookup Table Control Register. (EPDC\_AUTOWV\_LUT)

This register is used to access the waveform mode lookup table.

DATA -> AUTOWV\_LUT[ADDR] : Writing this reg with 'ADDR' and 'DATA' info will get 'DATA' written to AUTOWV\_LUT mem indexed with 'ADDR'

Address: 20F\_4000h base + 1C0h offset = 20F\_41C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								DATA								Reserved															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### EPDC\_AUTOWV\_LUT field descriptions

Field	Description
31–24 -	This field is reserved. Reserved, always set to zero.
23–16 DATA	DATA
15–3 -	This field is reserved. Reserved, always set to zero.
2–0 ADDR	ADDR

## 22.4.15 EPDC Timing Control Engine Control Register (EPDC\_TCE\_CTRLn)

TCE general control register

This register houses Horizontal scan timing. Note that line data length is derived from EPDC\_RES.

Address: 20F\_4000h base + 200h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								VSCAN_HOLDOFF							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				VCOM_VAL		VCOM_MODE	DDR_MODE	LVDS_MODE_	LVDS_MODE	SCAN_DIR_1	SCAN_DIR_0	DUAL_SCAN	SDDO_WIDTH	PIXELS_	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

**EPDC\_TCE\_CTRLn field descriptions**

Field	Description
31–25 -	This field is reserved. Reserved.
24–16 VSCAN_	This period (expressed in vertical lines), sets the portion of the vertical blanking available for new LUTs to be activated. The remainder of the blanking period is reserved for pre-filling the TCE pixel FIFOs. Increasing this value allows for multiple smaller updates to be intercepted by the current frame scan. This number should not exceed FRAME_END+FRAME_SYNC+FRAME_BEGIN. Increasing this value can improve the ability for any given update to intercept the next available frame-scan. Excessive values can result in TCE FIFO under-runs.
15–12 -	This field is reserved. Reserved.
11–10 VCOM_VAL	When VCOM_MODE = MANUAL, this value is used to manually set the VCOM value for the VCOM[1:0] pins
9 VCOM_MODE	This field determines the method used to drive the VCOM signal. 0x0 <b>MANUAL</b> — VCOM Value is set manually using VCOM_VAL field 0x1 <b>AUTO</b> — VCOM Value is used from waveform
8 DDR_MODE	If set, DATA is driven on both positive and negative edges of SDCLK. Note that this mode is not supported when SDDO_BUS_FORMAT=16BIT and LVDS is not used. This must always be set when LVDS_MODE is set.

Table continues on the next page...

## EPDC\_TCE\_CTRLn field descriptions (continued)

Field	Description
7 LVDS_MODE_CE	If set (together with LVDS_MODE=1), SDCE[9:5] shall be driven as the differential inverse of SDCE[4:0]. In this mode the EPDC only supports 5 CE lines.
6 LVDS_MODE	If set, the upper 8-bit of the DATA bus are used for LVDS differential signalling. Note that this can only be used when SDDO_BUS_FORMAT is set to 16BIT, i.e. LVDS signaling is not supported with an 8-bit DATA interface. Note that for LVDS_MODE, DDR_MODE must also be set.
5 SCAN_DIR_1	Determines scan direction for each half of the TFT panel 0x1 <b>UP</b> — Scan this region from bottom to top 0x0 <b>DOWN</b> — Scan this region from top to bottom
4 SCAN_DIR_0	Determines scan direction for each half of the TFT panel 0x1 <b>UP</b> — Scan this region from bottom to top 0x0 <b>DOWN</b> — Scan this region from top to bottom
3 DUAL_SCAN	Enables dual scan-mode. Note in this mode, SDDO_BUS_FORMAT=16BIT must be selected. and PIXELS_PER_CLK applies to each 8-bit segment of the DATA bus.
2 SDDO_WIDTH	Selects either 8 or 16 bit DATA bus format 0x0 <b>8BIT</b> — Connect to 8-bit source driver 0x1 <b>16BIT</b> — Connct to 16-bit source driver
1–0 PIXELS_PER_SDCLK	Number of TFT pixels per SDCLK period. Note that this value forms the division of the PIXLK to generate the SDCLK such that $SDCLK = PIXCLK / PIXELS\_PER\_SDCLK$ . For dual-scan mode (DUAL_SCAN=1), this applies to 8-bit half of the 16-bit DATA. It should be noted that when DDR_MODE is enabled, both edges of the clock have to be accounted for in this value, so for example with an 8-bit DATA, 2-bit TFT pixel, this field should be set to EIGHT (four pixels on the pos-edge and 4-pixels on the neg-edge) 0x0 <b>RESERVED</b> — Reserved 0x1 <b>TWO</b> — Two TFT-pixels per SDCLK 0x2 <b>FOUR</b> — Four TFT-pixels per SDCLK 0x3 <b>EIGHT</b> — Eight TFT-pixels per SDCLK



## 22.4.16 EPDC Timing Control Engine Source-Driver Config Register (EPDC\_TCE\_SDCFGn)

### Source-driver configuration register

Address: 20F\_4000h base + 220h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved										SDCLK_HOLD	SDSHR	NUM_CE			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SDDO_REFORMAT		SDDO_INVERT	PIXELS_PER_CE												
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### EPDC\_TCE\_SDCFGn field descriptions

Field	Description
31–22 -	This field is reserved. Reserved.
21 SDCLK_HOLD	Setting this bit to 1 holds the SDCLK low during LINE_BEGIN
20 SDSHR	Value for source-driver shift direction output port
19–16 NUM_CE	Number of source driver IC chip-enables. Must be 1-10
15–14 SDDO_REFORMAT	This register defines the various re-formatting options to enable more flexibility in the source-driver interface:  0x0 <b>STANDARD</b> — No change. 0x1 <b>FLIP_PIXELS</b> — Reverses the order of the pixels on DATA. This register setting is sensitive to the TFT pixel width (TFT_PIXEL_FORMAT), e.g. for TFT_PIXEL_FORMAT=2B on an 8-bit bus P3,P2,P1,P0 becomes P0,P1,P2,P3, whereas with TFT_PIXEL_FORMAT=4B, on an 8-bit bus, P1,P0 becomes P0,P1
13 SDDO_INVERT	Setting this bit to 1 reverses the polarity of each DATA bit so 0xAAAA in 16-bit mode for example becomes 0x5555. This setting can be made in addition to the SDDO_REFORMAT register setting.
12–0 PIXELS_PER_CE	Number of pixels (outputs) per source-driver IC. Please note that EPDC_RES[HORIZONTAL] must be an integer multiple of PINS_PER_CE.

## 22.4.17 EPDC Timing Control Engine Gate-Driver Config Register (EPDC\_TCE\_GDCFGn)

This register houses gate-driver configuration.

Address: 20F\_4000h base + 240h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PERIOD_VSCAN															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								GDRL				Reserved		GDOE_MODE	GDSP_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### EPDC\_TCE\_GDCFGn field descriptions

Field	Description
31–16 PERIOD_VSCAN	when vscan state is splited, this reg defines the counter period
15–5 -	This field is reserved. Reserved.
4 GDRL	Value for gate-driver right/left shift output port
3–2 -	This field is reserved. Reserved.
1 GDOE_MODE	Selects method for driving GDOE signal. When set to 0, GDOE is driven at all times during the frame-scan except FRAME_SYNC. When set to 1, GDOE is driven as a delayed version of GDCLK delayed by EPDC_TCE_TIMING3[GDOE_OFFSET].
0 GDSP_MODE	Selects method for driving GDSP pulse. When set to 0, GDSP is always fixed to have a pulse width of one line-time. When set to 1, GDSP has a pulse-width determined by the FRAME_SYNC setting. Note that GDSP_MODE=1 is not compatible with the GDSP_OFFSET function

## 22.4.18 EPDC Timing Control Engine Horizontal Timing Register 1 (EPDC\_TCE\_HSCAN1n)

Horizontal scan timing registers. Note that all timing values are expressed in terms of the EPDC's internal PIXCLK, which depending on the PIXELS\_PER\_SDCLK register setting is either 2:1 or 4:1

This register houses Horizontal scan timing. Note that line data length is derived from EPDC\_RES.

Address: 20F\_4000h base + 260h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				LINE_SYNC_WIDTH												Reserved				LINE_SYNC											
W	Reserved				LINE_SYNC_WIDTH												Reserved				LINE_SYNC											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_TCE\_HSCAN1n field descriptions

Field	Description
31–28 -	This field is reserved. Reserved.
27–16 LINE_SYNC_WIDTH	Number of PIXCLK cycles for the SDLE active time. Note that this value cannot be larger than LINE_SYNC and must be greater than 0. Typically it is recommended to set this value to be the same as LINE_SYNC
15–12 -	This field is reserved. Reserved.
11–0 LINE_SYNC	Number of PIXCLK cycles for line sync duration. Note that this value encompasses the LINE_SYNC_WIDTH duration. This value must be programmed to a multiple of SDCLK cycles

## 22.4.19 EPDC Timing Control Engine Horizontal Timing Register 2 (EPDC\_TCE\_HSCAN2n)

Horizontal scan timing registers. Note that all timing values are expressed in terms of the EPDC's internal PIXCLK, which depending on the PIXELS\_PER\_SDCLK register setting is either 2:1 or 4:1

This register houses Horizontal scan timing. Note that line data length is derived from EPDC\_RES.

Address: 20F\_4000h base + 280h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				LINE_END												Reserved				LINE_BEGIN											
W	Reserved				LINE_END												Reserved				LINE_BEGIN											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**EPDC\_TCE\_HSCAN2n field descriptions**

Field	Description
31–28 -	This field is reserved. Reserved.
27–16 LINE_END	Number of PIXCLK cycles for line end duration. This defines the duration from the de-assertion of SDCE and assertion of the next SDLE.
15–12 -	This field is reserved. Reserved.
11–0 LINE_BEGIN	Number of PIXCLK cycles for line begin duration. This defines the interval between de-assertion of SDLE and assertion of the SDCE signals. This value must be programmed to a multiple of SDCLK cycles

**22.4.20 EPDC Timing Control Engine Vertical Timing Register (EPDC\_TCE\_VSCANn)**

Vertical scan timing registers

This register houses vertical scan timing. Note that frame data length is derived from EPDC\_RES.

Address: 20F\_4000h base + 2A0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EPDC\_TCE\_VSCANn field descriptions**

Field	Description
31–24 -	This field is reserved. Reserved.
23–16 FRAME_END	Number of lines for frame end duration.
15–8 FRAME_BEGIN	Number of lines for frame begin duration.
7–0 FRAME_SYNC	Number of lines for frame sync duration.

**22.4.21 EPDC Timing Control Engine OE timing control Register (EPDC\_TCE\_OEn)**

This register contain delay programming values for the SDOEZ and SDOED source driver control signals

This register contain delay programming values for the SDOZ and SDOE source driver control signals

Address: 20F\_4000h base + 2C0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SDOED_WIDTH								SDOED_DLY								SDOEZ_WIDTH								SDOEZ_DLY							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_TCE\_OEn field descriptions

Field	Description
31–24 SDOED_WIDTH	Number of PIXCLK cycles from SDOED high to SDOED falling (Must be greater than 0)
23–16 SDOED_DLY	Number of PIXCLK cycles from SDOEZ low to SDOED rising (Must be greater than 0)
15–8 SDOEZ_WIDTH	Number of PIXCLK cycles from SDOEZ high to SDOEZ falling (Must be greater than 0)
7–0 SDOEZ_DLY	Number of PIXCLK cycles from SDLE falling edge to SDOEZ rising (Must be greater than 0)

## 22.4.22 EPDC Timing Control Engine Driver Polarity Register (EPDC\_TCE\_POLARITY<sub>n</sub>)

This registers allows for programming the polarity of source/gate driver control signals

This register houses FIFO control bits

Address: 20F\_4000h base + 2E0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0

**EPDC\_TCE\_POLARITY $n$  field descriptions**

Field	Description
31–5 -	This field is reserved. Reserved.
4 GDSP_POL	0 = Active Low, 1 = Active High. Applies to the GDSP output
3 GDOE_POL	0 = Active Low, 1 = Active High. Applies to the GDOE output
2 SDOE_POL	0 = Active Low, 1 = Active High. Applies to the SDOE. Does not apply to SDOEZ and SDOED outputs
1 SDLE_POL	0 = Active Low, 1 = Active High. Applies to the SDLE output
0 SDCE_POL	0 = Active Low, 1 = Active High. Applies to all 10 SDCE outputs

**22.4.23 EPDC Timing Control Engine Timing Register 1 (EPDC\_TCE\_TIMING1 $n$ )**

This register contains various timing adjustment controls

This register houses general purpose timing adjustment registers

Address: 20F\_4000h base + 300h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								SDLE_SHIFT		SDCLK_INVERT		Reserved	SDCLK_SHIFT		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EPDC\_TCE\_TIMING1n field descriptions**

Field	Description
31–6 -	This field is reserved. Reserved.
5–4 SDLE_SHIFT	This register can be used to implement additional timing setup/hold adjustment of source driver signals by adjusting the SDCLK up to 3 PIXCLK cycles  0x0 <b>NONE</b> — No shift of SDLE 0x1 <b>ONE</b> — Shift SDLE 1 pixclk cycle 0x2 <b>TWO</b> — Shift SDLE 2 pixclk cycles 0x3 <b>THREE</b> — Shift SDLE 3 pixclk cycles
3 SDCLK_INVERT	Invert phase of SDCLK
2 -	This field is reserved. Reserved.
1–0 SDCLK_SHIFT	This register can be used to implement additional timing setup/hold adjustment of source driver signals by adjusting the SDCLK up to 4 cycles  0x0 <b>NONE</b> — No shift of SDCLK 0x1 <b>ONE</b> — Shift SDCLK 1 pixclk cycle 0x2 <b>TWO</b> — Shift SDCLK 2 pixclk cycles 0x3 <b>THREE</b> — Shift SDCLK 3 pixclk cycles

**22.4.24 EPDC Timing Control Engine Timing Register 2 (EPDC\_TCE\_TIMING2n)**

This register contains various timing adjustment controls

This register houses general purpose timing adjustment registers

Address: 20F\_4000h base + 310h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GDCLK_HP																GDSP_OFFSET															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	

**EPDC\_TCE\_TIMING2n field descriptions**

Field	Description
31–16 GDCLK_HP	This register controls the GDCLK high-pulse width. It is expressed by N PIXCLKs where N=1 to 65535. Note that GDCLK will always have a period equal to the line-clock timing. A value of 0 is not supported. It is recommended that this value be set to at least a half line-clock time. For panels which use GDCLK to drive GDOE, this high-pulse width should be set to cover the majority of the line timing
15–0 GDSP_OFFSET	This register allows the user to shift the GDSP pulse by N PIXCLKs where N=1 to 65535. Note that GDSP will always have a pulse width equivalent to the line-clock timing. A value of 0 is not supported.

## 22.4.25 EPDC Timing Control Engine Timing Register 3 (EPDC\_TCE\_TIMING3n)

This register contains various timing adjustment controls

This register houses general purpose timing adjustment registers

Address: 20F\_4000h base + 320h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	GDOE_OFFSET																GDCLK_OFFSET																
W																																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1		

### EPDC\_TCE\_TIMING3n field descriptions

Field	Description
31–16 GDOE_OFFSET	When using GDOE_MODE=1, this register sets the delay from GDCLK to the GDOE in terms of N PIXCLK cycles
15–0 GDCLK_OFFSET	This register allows the user to shift the GDCLK from the line time by N PIXCLK cycles.

## 22.4.26 EPDC Pigeon Mode Control Register 0 (EPDC\_PIGEON\_CTRL0n)

This register contains global counter settings for Pigeon Mode

This register houses general purpose timing adjustment registers

Address: 20F\_4000h base + 380h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				LD_PERIOD												Reserved				FD_PERIOD											
W	Reserved				LD_PERIOD												Reserved				FD_PERIOD											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_PIGEON\_CTRL0n field descriptions

Field	Description
31–28 -	This field is reserved. Reserved.
27–16 LD_PERIOD	period of pclk counter during LD phase
15–12 -	This field is reserved. Reserved.
11–0 FD_PERIOD	period of line counter during FD phase



## 22.4.27 EPDC Pigeon Mode Control Register 1 (EPDC\_PIGEON\_CTRL1n)

This register contains global counter setting for pigeon mode

This register houses general purpose timing adjustment registers

Address: 20F\_4000h base + 390h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				FRAME_CNT_CYCLES												Reserved				FRAME_CNT_PERIOD											
W	Reserved				FRAME_CNT_CYCLES												Reserved				FRAME_CNT_PERIOD											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### EPDC\_PIGEON\_CTRL1n field descriptions

Field	Description
31–28 -	This field is reserved. Reserved.
27–16 FRAME_CNT_ CYCLES	max cycles of frame counter
15–12 -	This field is reserved. Reserved.
11–0 FRAME_CNT_ PERIOD	period of frame counter

## 22.4.28 EPDC IRQ Mask Register for LUT 0~31 (EPDC\_IRQ\_MASK1n)

Controls masking EPDC LUT complete interrupts

This register controls LUT0~31 IRQ masks for EPDC interrupts

Address: 20F\_4000h base + 3C0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LUTN_CMPLT_IRQ_EN																															
W	LUTN_CMPLT_IRQ_EN																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### EPDC\_IRQ\_MASK1n field descriptions

Field	Description
31–0 LUTN_CMPLT_ IRQ_EN	LUT0~31 Complete Interrupt Enable

## 22.4.29 EPDC IRQ Mask Register for LUT 32~63 (EPDC\_IRQ\_MASK2n)

Controls masking EPDC LUT complete interrupts

This register controls LUT0~31 IRQ masks for EPDC interrupts

Address: 20F\_4000h base + 3D0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LUTN_CMPLT_IRQ_EN																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### EPDC\_IRQ\_MASK2n field descriptions

Field	Description
31–0 LUTN_CMPLT_IRQ_EN	LUT32~64 Complete Interrupt Enable

## 22.4.30 EPDC Interrupt Register for LUT 0~31 (EPDC\_IRQ1n)

EPDC LUT Completion IRQs. The IRQ for a specific LUT is triggered when it's corresponding physical update is completed on the screen. Each interrupt has a corresponding mask register in EPDC\_IRQ\_MASK

This register houses the interrupt bits for the LUT Completions

Address: 20F\_4000h base + 3E0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LUTN_CMPLT_IRQ																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### EPDC\_IRQ1n field descriptions

Field	Description
31–0 LUTN_CMPLT_IRQ	LUT 0~31 Complete Interrupt

### 22.4.31 EPDC Interrupt Register for LUT 32~63 (EPDC\_IRQ2n)

EPDC LUT Completion IRQs. The IRQ for a specific LUT is triggered when it's corresponding physical update is completed on the screen. Each interrupt has a corresponding mask register in EPDC\_IRQ\_MASK

This register houses the interrupt bits for the LUT Completions

Address: 20F\_4000h base + 3F0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LUTN_CMPLT_IRQ																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### EPDC\_IRQ2n field descriptions

Field	Description
31–0 LUTN_CMPLT_IRQ	LUT 32~64 Complete Interrupt

### 22.4.32 EPDC IRQ Mask Register (EPDC\_IRQ\_MASKn)

Controls masking for all EPDC interrupts

This register controls IRQ masks for all EPDC interrupts

Address: 20F\_4000h base + 400h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								PWR_IRQ_EN	UPD_DONE_IRQ_EN	TCE_IDLE_IRQ_EN	BUS_ERROR_IRQ_EN	FRAME_END_IRQ_EN	TCE_UNDEERRUN_IRQ_EN	COL_IRQ_EN	WB_CMPLT_IRQ_EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## EPDC\_IRQ\_MASKn field descriptions

Field	Description
31–24 -	This field is reserved. Reserved.
23 PWR_IRQ_EN	Enable power interrupt
22 UPD_DONE_IRQ_EN	Enable UPD complete interrupt
21 TCE_IDLE_IRQ_EN	Enable TCE Idle interrupt detection.
20 BUS_ERROR_IRQ_EN	Enable AXI BUS ERROR interrupt detection.
19 FRAME_END_IRQ_EN	If this bit is set, EPDC will assert the current frame end interrupt. This irq is only available during updating period.
18 TCE_UNDERRUN_IRQ_EN	Enable pixel FIFO under-run condition detection.
17 COL_IRQ_EN	Enable collision detection interrupts for all LUTs
16 WB_CMPLT_IRQ_EN	Enable WB complete interrupt
15–0 -	This field is reserved. Reserved.

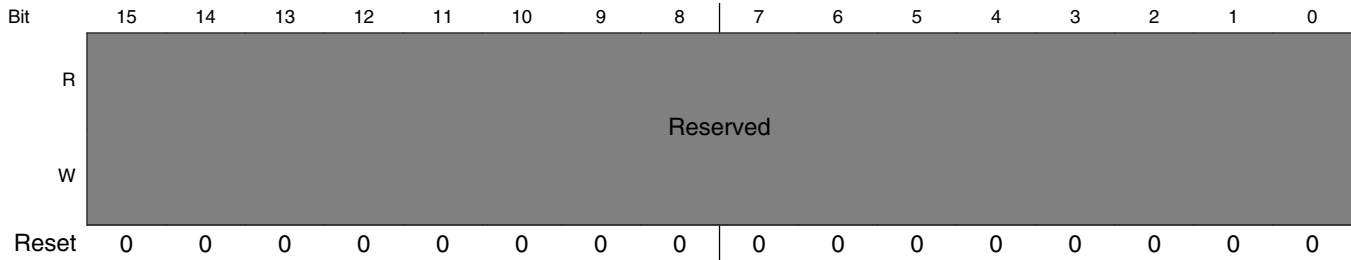
## 22.4.33 EPDC Interrupt Register (EPDC\_IRQn)

EPDC LUT Completion IRQs. The IRQ for a specific LUT is triggered when it's corresponding physical update is completed on the screen. Each interrupt has a corresponding mask register in EPDC\_IRQ\_MASK

This register houses the interrupt bits for the LUT Completions

Address: 20F\_4000h base + 420h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								PWR_IRQ	UPD_DONE_IRQ	TCE_IDLE_IRQ	BUS_ERROR_IRQ	FRAME_END_IRQ	TCE_UNDERRUN_IRQ	LUT_COL_IRQ	WB_CMPLT_IRQ
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



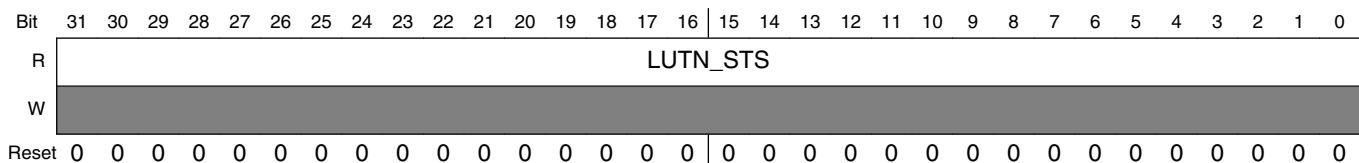
### EPDC\_IRQn field descriptions

Field	Description
31–24 -	This field is reserved. Reserved.
23 PWR_IRQ	Power Interrupt
22 UPD_DONE_IRQ	Working buffer process complete Interrupt
21 TCE_IDLE_IRQ	Interrupt to indicate that the TCE has completed TFT frame scans and is in an idle state.
20 BUS_ERROR_IRQ	Interrupt to indicate AXI BUS error occurs.
19 FRAME_END_IRQ	Interrupt to indicate EPDC has completed the current frame and is in the vertical blanking period.
18 TCE_UNDERRUN_IRQ	Interrupt to indicate that a pixel FIFO under-run has occurred.
17 LUT_COL_IRQ	Collision detection interrupt. Check EPDC_STATUS_COL.
16 WB_CMPLT_IRQ	Working buffer process complete Interrupt
15–0 -	This field is reserved. Reserved.

## 22.4.34 EPDC Status Register - LUTs (EPDC\_STATUS\_LUTS1n)

### EPDC Status Register - LUTS 0~31

Address: 20F\_4000h base + 440h offset + (4d × i), where i=0d to 3d



**EPDC\_STATUS\_LUTS1*n* field descriptions**

Field	Description
31–0 LUTN_STS	LUT 0~31 Status : 1=ACTIVE, 0=IDLE

**22.4.35 EPDC Status Register - LUTs (EPDC\_STATUS\_LUTS2*n*)****EPDC Status Register - LUTS 0~31**

Address: 20F\_4000h base + 450h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LUTN_STS																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EPDC\_STATUS\_LUTS2*n* field descriptions**

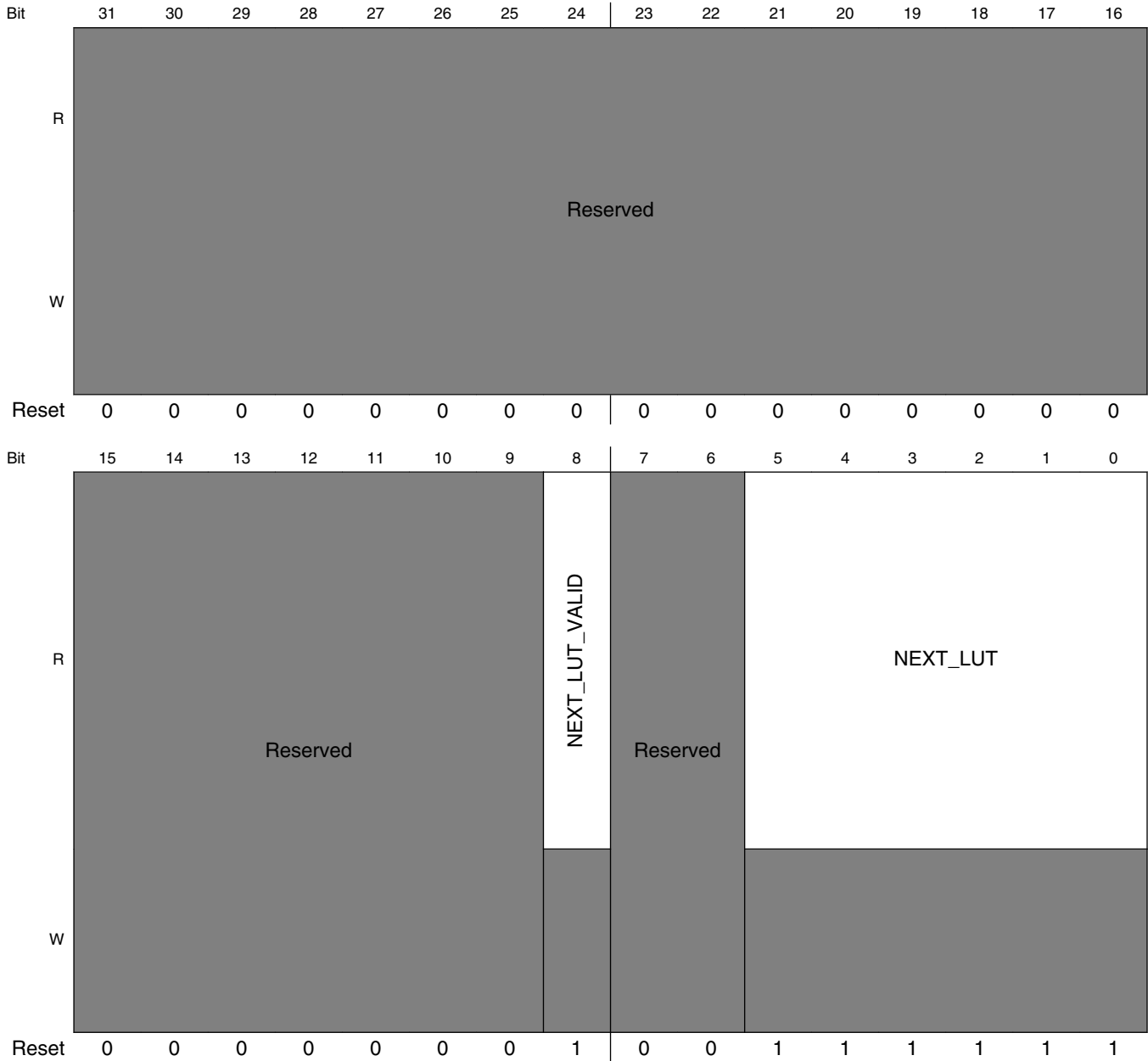
Field	Description
31–0 LUTN_STS	LUT 32~63 Status : 1=ACTIVE, 0=IDLE

**22.4.36 EPDC Status Register - Next Available LUT (EPDC\_STATUS\_NEXTLUT)**

Holds value of next available LUT. Can be used for fast LUT assignment. This value can be read and then used in an update command as part of the EPDC\_UPD\_CTRL register write

The DIGCTL Status Register provides a read-only view to various input conditions and internal states.

Address: 20F\_4000h base + 460h offset = 20F\_4460h

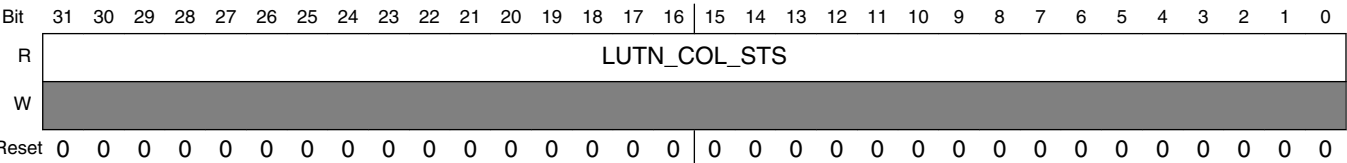
**EPDC\_STATUS\_NEXTLUT field descriptions**

Field	Description
31–9 -	This field is reserved. Reserved.
8 NEXT_LUT_ VALID	This bitfield can be used to check against a LUTs full condition
7–6 -	This field is reserved. Reserved.
5–0 NEXT_LUT	Next available LUT value

22.4.37 EPDC LUT Collision Status (EPDC\_STATUS\_COL1n)

EPDC LUT Collision Status Register and works in conjunction with EPDC\_IRQ[LUT\_COL\_IRQ]. When a collision occurs the interrupt is set and all status bits are set for LUTs which were touched by the collision. It does not set the bit for the LUT which caused the collision. There is a single interrupt mask which is used to control all the IRQ bits in this register (in EPDC\_IRQ\_MASK). Note that a collision caused by a LUT which was set-up for no collision detection will not trigger any collision LUT IRQ or status update. Note that clearing the interrupt bit EPDC\_IRQ[LUT\_COL\_IRQ] clears this register

Address: 20F\_4000h base + 480h offset + (4d × i), where i=0d to 3d



EPDC\_STATUS\_COL1n field descriptions

Field	Description
31–0 LUTN_COL_STS	LUTn Collision Status



### 22.4.38 EPDC LUT Collision Status (EPDC\_STATUS\_COL2n)

EPDC LUT Collision Status Register and works in conjunction with EPDC\_IRQ[LUT\_COL\_IRQ]. When a collision occurs the interrupt is set and all status bits are set for LUTs which were touched by the collision. It does not set the bit for the LUT which caused the collision. There is a single interrupt mask which is used to control all the IRQ bits in this register (in EPDC\_IRQ\_MASK). Note that a collision caused by a LUT which was set-up for no collision detection will not trigger any collision LUT IRQ or status update. Note that clearing the interrupt bit EPDC\_IRQ[LUT\_COL\_IRQ] clears this register

Address: 20F\_4000h base + 490h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LUTN_COL_STS																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EPDC\_STATUS\_COL2n field descriptions**

Field	Description
31–0 LUTN_COL_STS	LUTn Collision Status

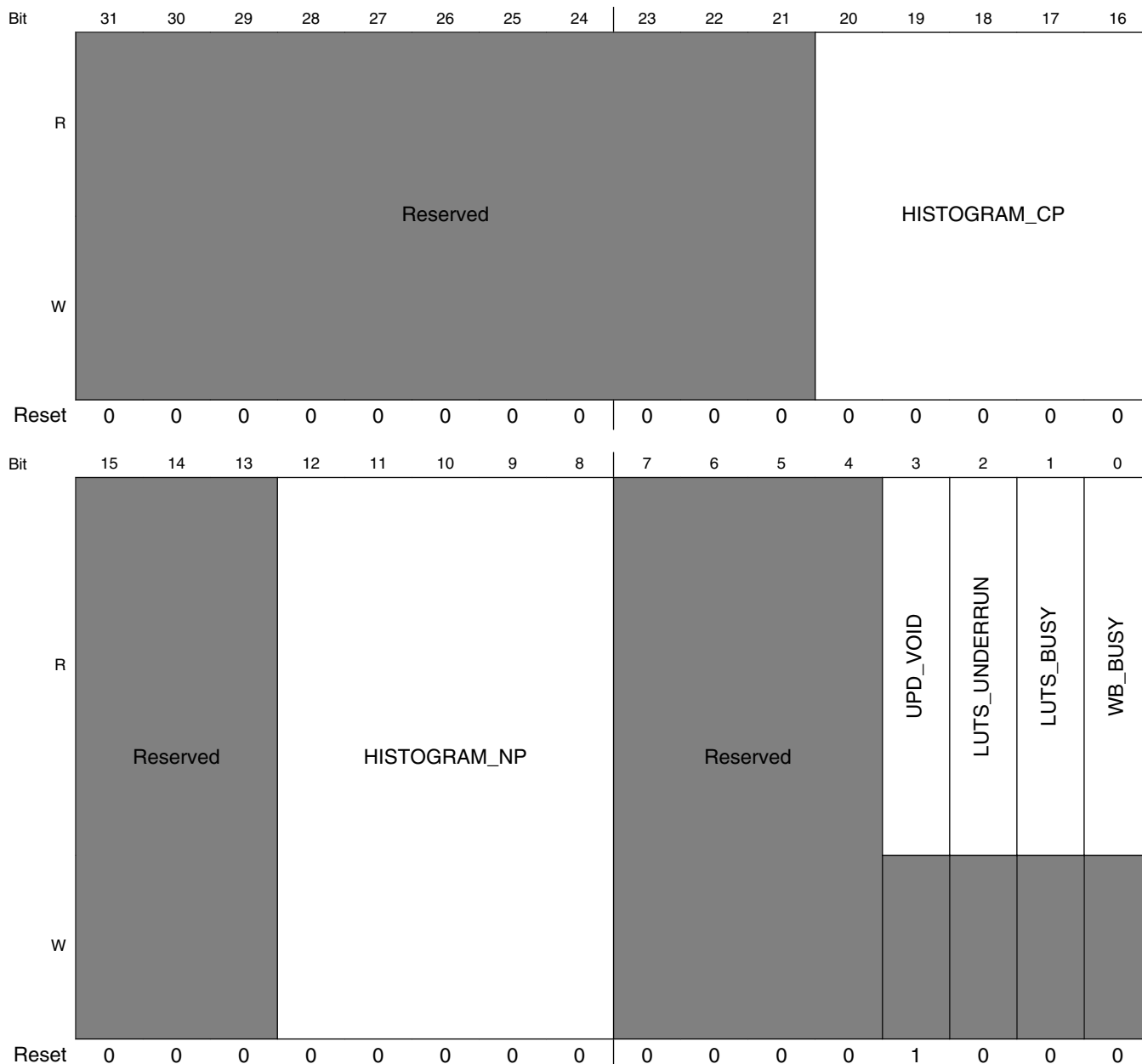
### 22.4.39 EPDC General Status Register (EPDC\_STATUSn)

Register to house non LUT specific status bits

This register houses general status bits

## EPDC Memory Map/Register Definition

Address: 20F\_4000h base + 4A0h offset + (4d × i), where i=0d to 3d



### EPDC\_STATUSn field descriptions

Field	Description
31–21 -	This field is reserved. Reserved.
20–16 HISTOGRAM_ CP	Indicates which histogram matched the existing bitmap(CP). Bit[0] indicates that the bitmap pixels were fully contained within the HIST1 (single color ) histogram. Bit[1] indicates that the bitmap pixels were fully contained within the HIST2 (black / white ) histogram. Bit[2] indicates that the bitmap pixels were fully contained within the HIST4 (2-bit grayscale) histogram. Bit[3] indicates that the bitmap pixels were fully contained within the HIST8 (3-bit grayscale) histogram.

*Table continues on the next page...*

**EPDC\_STATUSn field descriptions (continued)**

Field	Description
	Bit[4] indicates that the bitmap pixels were fully contained within the HIST16 (4-bit grayscale) histogram.
15–13 -	This field is reserved. Reserved.
12–8 HISTOGRAM_ NP	Indicates which histogram matched the processed bitmap(NP). Bit[0] indicates that the bitmap pixels were fully contained within the HIST1 (single color ) histogram. Bit[1] indicates that the bitmap pixels were fully contained within the HIST2 (black / white ) histogram. Bit[2] indicates that the bitmap pixels were fully contained within the HIST4 (2-bit grayscale) histogram. Bit[3] indicates that the bitmap pixels were fully contained within the HIST8 (3-bit grayscale) histogram. Bit[4] indicates that the bitmap pixels were fully contained within the HIST16 (4-bit grayscale) histogram.
7–4 -	This field is reserved. Reserved.
3 UPD_VOID	shows the update buffer is void (don't need any real pixel update)
2 LUTS_ UNDERRUN	Provides a summary status of LUT fill. 1= not enough time for active luts read during blanking period before vscan_holdoff. 0=complete all active luts fill during blanking period before VSCAN_HOLDOFF.
1 LUTS_BUSY	Provides a summary status of LUTs. 1= All LUTs are busy, 0= LUTs are available
0 WB_BUSY	Working buffer process is busy cannot accept new update requests. When WB_BUSY is 1, software should wait for the WB_CMPLT_IRQ interrupt. When this interrupt occurs WB_BUSY is cleared immediately. This is a real-time status of the process.

## 22.4.40 EPDC Collision Region Co-ordinate (EPDC\_UPD\_COL\_CORD)

EPDC Collision Region Co-ordinate, cleared when new update issued

This register only valid after WB completion and collision happens.

Address: 20F\_4000h base + 4C0h offset = 20F\_44C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved			YCORD													Reserved			XCORD												
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EPDC\_UPD\_COL\_CORD field descriptions**

Field	Description
31–29 -	This field is reserved. Reserved.
28–16 YCORD	Y co-ordinate for collision region of the latest completed update

*Table continues on the next page...*

**EPDC\_UPD\_COL\_CORD field descriptions (continued)**

Field	Description
15–13 -	This field is reserved. Reserved.
12–0 XCORD	X co-ordinate for collision region of the latest completed update

**22.4.41 EPDC Collision Region Size (EPDC\_UPD\_COL\_SIZE)**

EPDC Collision Region Size of the latest completed update cleared when new update issued

This register only valid after WB completion and collision happens.

Address: 20F\_4000h base + 4E0h offset = 20F\_44E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved			HEIGHT													Reserved			WIDTH												
W	Reserved																Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EPDC\_UPD\_COL\_SIZE field descriptions**

Field	Description
31–29 -	This field is reserved. Reserved.
28–16 HEIGHT	Height (in pixels)
15–13 -	This field is reserved. Reserved.
12–0 WIDTH	Width (in pixels)

**22.4.42 1-level Histogram Parameter Register.  
(EPDC\_HIST1\_PARAM)**

This register specifies the valid values for a 1-level(single color) histogram. If all pixels in a bitmap is only one color, STATUS[0] will be set at the end of frame processing. All comparator values should be programmed such that they are consistent with the TFT\_PIXEL\_FORMAT control field.

Address: 20F\_4000h base + 600h offset = 20F\_4600h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_HIST1\_PARAM field descriptions

Field	Description
31–5 RSVD	This field is reserved. Reserved, always set to zero.
4–0 VALUE0	value for 1-level histogram

## 22.4.43 2-level Histogram Parameter Register. (EPDC\_HIST2\_PARAM)

This register specifies the valid values for a 2-level histogram. If all pixels in a bitmap match these two values, STATUS[0] will be set at the end of frame processing. All comparator values should be programmed such that they are consistent with the TFT\_PIXEL\_FORMAT control field.

Address: 20F\_4000h base + 610h offset = 20F\_4610h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																Reserved			VALUE1				Reserved			VALUE0					
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	

### EPDC\_HIST2\_PARAM field descriptions

Field	Description
31–16 RSVD	This field is reserved. Reserved, always set to zero.
15–13 -	This field is reserved. Reserved, always set to zero.
12–8 VALUE1	White value for 2-level histogram
7–5 -	This field is reserved. Reserved, always set to zero.
4–0 VALUE0	Black value for 2-level histogram

## 22.4.44 4-level Histogram Parameter Register. (EPDC\_HIST4\_PARAM)

This register specifies the valid values for a 4-level histogram. If all pixels in a bitmap match these two values, STATUS[1] will be set at the end of frame processing. All comparator values should be programmed such that they are consistent with the TFT\_PIXEL\_FORMAT control field.

Address: 20F\_4000h base + 620h offset = 20F\_4620h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																																
R	Reserved								VALUE3								Reserved								VALUE2								Reserved								VALUE1								Reserved								VALUE0							
W	Reserved								VALUE3								Reserved								VALUE2								Reserved								VALUE1								Reserved								VALUE0							
Reset	0	0	0	0	1	1	1	1	0	0	0	0	1	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0																															

### EPDC\_HIST4\_PARAM field descriptions

Field	Description
31–29 -	This field is reserved. Reserved, always set to zero.
28–24 VALUE3	GRAY3 (White) value for 4-level histogram
23–21 -	This field is reserved. Reserved, always set to zero.
20–16 VALUE2	GRAY2 value for 4-level histogram
15–13 -	This field is reserved. Reserved, always set to zero.
12–8 VALUE1	GRAY1 value for 4-level histogram
7–5 -	This field is reserved. Reserved, always set to zero.
4–0 VALUE0	GRAY0 (Black) value for 4-level histogram

## 22.4.45 8-level Histogram Parameter 0 Register. (EPDC\_HIST8\_PARAM0)

This register specifies four of the valid values for an 8-level histogram. If all pixels in a bitmap match these two values, STATUS[2] will be set at the end of frame processing. All comparator values should be programmed such that they are consistent with the TFT\_PIXEL\_FORMAT control field.

Address: 20F\_4000h base + 630h offset = 20F\_4630h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0																				
R	Reserved								VALUE3								Reserved				VALUE2								Reserved				VALUE1								Reserved				VALUE0							
W																																																				
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0																				

**EPDC\_HIST8\_PARAM0 field descriptions**

Field	Description
31–29 -	This field is reserved. Reserved, always set to zero.
28–24 VALUE3	GRAY3 value for 8-level histogram
23–21 -	This field is reserved. Reserved, always set to zero.
20–16 VALUE2	GRAY2 value for 8-level histogram
15–13 -	This field is reserved. Reserved, always set to zero.
12–8 VALUE1	GRAY1 value for 8-level histogram
7–5 -	This field is reserved. Reserved, always set to zero.
4–0 VALUE0	GRAY0 (Black) value for 8-level histogram

## 22.4.46 8-level Histogram Parameter 1 Register. (EPDC\_HIST8\_PARAM1)

This register specifies four of the valid values for an 8-level histogram. If all pixels in a bitmap match these two values, STATUS[2] will be set at the end of frame processing. All comparator values should be programmed such that they are consistent with the TFT\_PIXEL\_FORMAT control field.

Address: 20F\_4000h base + 640h offset = 20F\_4640h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
R	Reserved								VALUE7				Reserved				VALUE6				Reserved				VALUE5				Reserved				VALUE4			
W	Reserved								VALUE7				Reserved				VALUE6				Reserved				VALUE5				Reserved				VALUE4			
Reset	0	0	0	0	1	1	1	1	0	0	0	0	1	1	0	1	0	0	0	0	1	0	1	1	0	0	0	0	1	0	0	1				

**EPDC\_HIST8\_PARAM1 field descriptions**

Field	Description
31–29 -	This field is reserved. Reserved, always set to zero.

*Table continues on the next page...*

**EPDC\_HIST8\_PARAM1 field descriptions (continued)**

Field	Description
28–24 VALUE7	GRAY7 (White) value for 8-level histogram
23–21 -	This field is reserved. Reserved, always set to zero.
20–16 VALUE6	GRAY6 value for 8-level histogram
15–13 -	This field is reserved. Reserved, always set to zero.
12–8 VALUE5	GRAY5 value for 8-level histogram
7–5 -	This field is reserved. Reserved, always set to zero.
4–0 VALUE4	GRAY4 value for 8-level histogram

## 22.4.47 16-level Histogram Parameter 0 Register. (EPDC\_HIST16\_PARAM0)

This register specifies four of the valid values for a 16-level histogram. If all pixels in a bitmap match these two values, STATUS[3] will be set at the end of frame processing. All comparator values should be programmed such that they are consistent with the TFT\_PIXEL\_FORMAT control field.

Address: 20F\_4000h base + 650h offset = 20F\_4650h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

**EPDC\_HIST16\_PARAM0 field descriptions**

Field	Description
31–29 -	This field is reserved. Reserved, always set to zero.
28–24 VALUE3	GRAY3 value for 16-level histogram
23–21 -	This field is reserved. Reserved, always set to zero.
20–16 VALUE2	GRAY2 value for 16-level histogram
15–13 -	This field is reserved. Reserved, always set to zero.

*Table continues on the next page...*



**EPDC\_HIST16\_PARAM0 field descriptions (continued)**

Field	Description
12–8 VALUE1	GRAY1 value for 16-level histogram
7–5 -	This field is reserved. Reserved, always set to zero.
4–0 VALUE0	GRAY0 (Black) value for 16-level histogram

### 22.4.48 16-level Histogram Parameter Register. (EPDC\_HIST16\_PARAM1)

This register specifies four of the valid values for a 16-level histogram. If all pixels in a bitmap match these two values, STATUS[3] will be set at the end of frame processing. All comparator values should be programmed such that they are consistent with the TFT\_PIXEL\_FORMAT control field.

Address: 20F\_4000h base + 660h offset = 20F\_4660h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0

**EPDC\_HIST16\_PARAM1 field descriptions**

Field	Description
31–29 -	This field is reserved. Reserved, always set to zero.
28–24 VALUE7	GRAY7 value for 16-level histogram
23–21 -	This field is reserved. Reserved, always set to zero.
20–16 VALUE6	GRAY6 value for 16-level histogram
15–13 -	This field is reserved. Reserved, always set to zero.
12–8 VALUE5	GRAY5 value for 16-level histogram
7–5 -	This field is reserved. Reserved, always set to zero.
4–0 VALUE4	GRAY4 value for 16-level histogram

## 22.4.49 16-level Histogram Parameter Register. (EPDC\_HIST16\_PARAM2)

This register specifies four of the valid values for a 16-level histogram. If all pixels in a bitmap match these two values, STATUS[3] will be set at the end of frame processing. All comparator values should be programmed such that they are consistent with the TFT\_PIXEL\_FORMAT control field.

Address: 20F\_4000h base + 670h offset = 20F\_4670h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	1	0	1	1	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	1	0	0	0	0	1	0	0	0

**EPDC\_HIST16\_PARAM2 field descriptions**

Field	Description
31–29 -	This field is reserved. Reserved, always set to zero.
28–24 VALUE11	GRAY11 value for 16-level histogram
23–21 -	This field is reserved. Reserved, always set to zero.
20–16 VALUE10	GRAY10 value for 16-level histogram
15–13 -	This field is reserved. Reserved, always set to zero.
12–8 VALUE9	GRAY9 value for 16-level histogram
7–5 -	This field is reserved. Reserved, always set to zero.
4–0 VALUE8	GRAY8 value for 16-level histogram

## 22.4.50 16-level Histogram Parameter Register. (EPDC\_HIST16\_PARAM3)

This register specifies four of the valid values for a 16-level histogram. If all pixels in a bitmap match these two values, STATUS[3] will be set at the end of frame processing. All comparator values should be programmed such that they are consistent with the TFT\_PIXEL\_FORMAT control field.

Address: 20F\_4000h base + 680h offset = 20F\_4680h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
R	Reserved								VALUE15				Reserved				VALUE14				Reserved				VALUE13				Reserved				VALUE12			
W	Reserved								VALUE15				Reserved				VALUE14				Reserved				VALUE13				Reserved				VALUE12			
Reset	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	0	0	0	0	1	1	0	1	0	0	0	0	1	1	0	0					

### EPDC\_HIST16\_PARAM3 field descriptions

Field	Description
31–29 -	This field is reserved. Reserved, always set to zero.
28–24 VALUE15	GRAY15 (White) value for 16-level histogram
23–21 -	This field is reserved. Reserved, always set to zero.
20–16 VALUE14	GRAY14 value for 16-level histogram
15–13 -	This field is reserved. Reserved, always set to zero.
12–8 VALUE13	GRAY13 value for 16-level histogram
7–5 -	This field is reserved. Reserved, always set to zero.
4–0 VALUE12	GRAY12 value for 16-level histogram

## 22.4.51 EPDC General Purpose I/O Debug register (EPDC\_GPION)

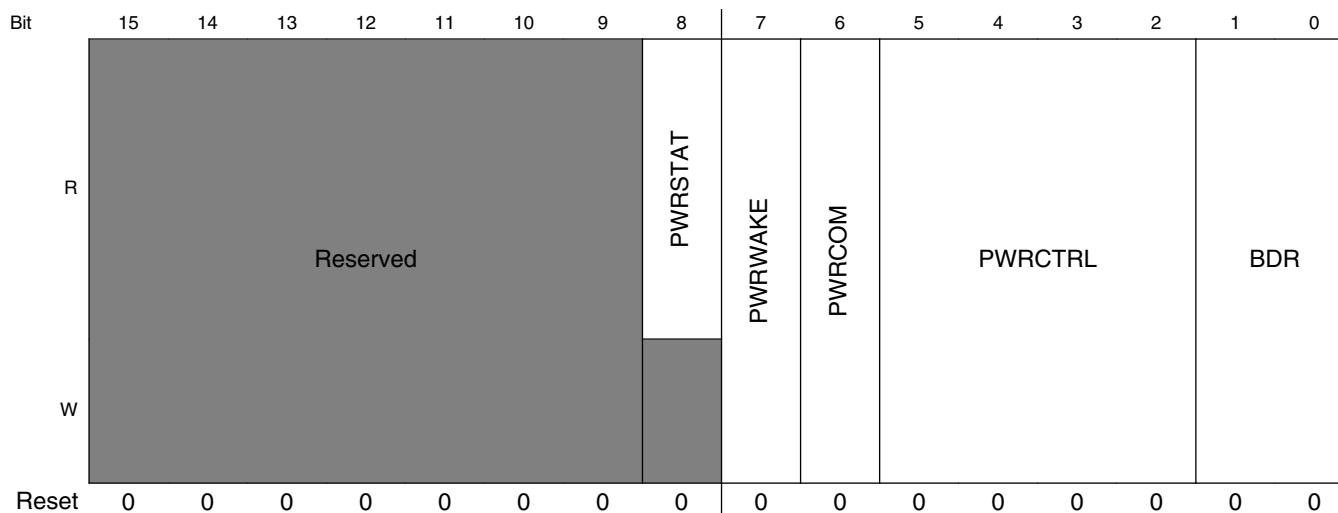
GPIO register to control EPDC\_BDR[1:0], EPDC\_PWR\_CTRL[3:0] and EPDC\_PWR\_COM output signals

Houses software control signal registers

Address: 20F\_4000h base + 700h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## EPDC Memory Map/Register Definition



### EPDC\_GPIOn field descriptions

Field	Description
31–9 -	This field is reserved. Reserved.
8 PWRSTAT	reflect ipp_epdc_pwrstat input
7 PWRWAKE	Controls ipp_epdc_pwrwake output
6 PWRCOM	Controls ipp_epdc_pwrcom output
5–2 PWRCTRL	Controls ipp_epdc_pwrctrl[3:0] output
1–0 BDR	Controls ipp_epdc_bdr[1:0] output

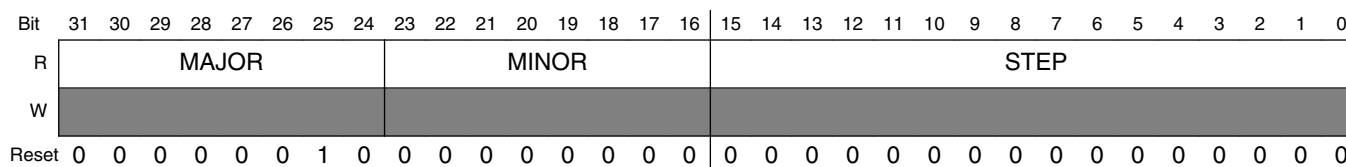
## 22.4.52 EPDC Version Register (EPDC\_VERSION)

This register reflects the version number for the EPDC.

### EXAMPLE

No Example.

Address: 20F\_4000h base + 7F0h offset = 20F\_47F0h



**EPDC\_VERSION field descriptions**

Field	Description
31–24 MAJOR	Fixed read-only value reflecting the MAJOR field of the RTL version.
23–16 MINOR	Fixed read-only value reflecting the MINOR field of the RTL version.
15–0 STEP	Fixed read-only value reflecting the stepping of the RTL version.

## 22.4.53 Panel Interface Signal Generator Register 0\_0 (EPDC\_PIGEON\_0\_0)

parameters for timing signal generation

Address: 20F\_4000h base + 800h offset = 20F\_4800h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATE_MASK								MASK_CNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MASK_CNT				MASK_CNT_SEL				OFFSET				INC_SEL		POL	EN
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

**EPDC\_PIGEON\_0\_0 field descriptions**

Field	Description
31–24 STATE_MASK	state_mask = (FSIFBIFDIFE) and (LSILBILDILE) , select any combination of scan states as reference point for local counter to start ticking  0x1 <b>FS</b> — FRAME SYNC 0x2 <b>FB</b> — FRAME BEGIN 0x4 <b>FD</b> — FRAME DATA 0x8 <b>FE</b> — FRAME END 0x10 <b>LS</b> — LINE SYNC 0x20 <b>LB</b> — LINE BEGIN 0x40 <b>LD</b> — LINE DATA 0x80 <b>LE</b> — LINE END
23–12 MASK_CNT	When the global counter selected through MASK_CNT_SEL matches value in this reg, pigeon local counter start ticking. 0=disable
11–8 MASK_CNT_SEL	select global counters as mask condition, use together with MASK_CNT  0x0 <b>HSTATE_CNT</b> — pclk counter within one hscan state 0x1 <b>HSTATE_CYCLE</b> — pclk cycle within one hscan state 0x2 <b>VSTATE_CNT</b> — line counter within one vscan state 0x3 <b>VSTATE_CYCLE</b> — line cycle within one vscan state

*Table continues on the next page...*

**EPDC\_PIGEON\_0\_0 field descriptions (continued)**

Field	Description
	0x4 <b>FRAME_CNT</b> — frame counter 0x5 <b>FRAME_CYCLE</b> — frame cycle 0x6 <b>HCNT</b> — horizontal counter (pclk counter within one line ) 0x7 <b>VCNT</b> — vertical counter (line counter within one frame)
7–4 OFFSET	offset on pclk unit. 0=align with data, positive value means delay, minus value mean ahead. Supported range depends on panel mode
3–2 INC_SEL	event to increment local counter  0x0 <b>PCLK</b> — pclk 0x1 <b>LINE</b> — line start pulse 0x2 <b>FRAME</b> — frame start pulse 0x3 <b>SIG_ANOTHER</b> — use another signal as tick event
1 POL	polarity of signal output  0x0 <b>ACTIVE_HIGH</b> — normal signal (active high) 0x1 <b>ACTIVE_LOW</b> — inverted signal (active low)
0 EN	enable pigeon mode on this signal

## 22.4.54 Panel Interface Signal Generator Register 0\_1 (EPDC\_PIGEON\_0\_1)

parameters for timing signal generation

Address: 20F\_4000h base + 810h offset = 20F\_4810h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLR_CNT																SET_CNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**EPDC\_PIGEON\_0\_1 field descriptions**

Field	Description
31–16 CLR_CNT	deassert signal output when counter match this value  0x0 <b>CLEAR_USING_MASK</b> — keep active until mask off
15–0 SET_CNT	assert signal output when counter match this value  0x0 <b>START_ACTIVE</b> — start as active

## 22.4.55 Panel Interface Signal Generator Register 0\_1 (EPDC\_PIGEON\_0\_2)

parameters for timing signal generation

Address: 20F\_4000h base + 820h offset = 20F\_4820h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																SIG_ANOTHER						SIG_LOGIC									
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_PIGEON\_0\_2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved.
8–4 SIG_ANOTHER	select another signal for logic operation or as mask or counter tick event
3–0 SIG_LOGIC	logic operation with another signal sigout : final output signal of this generator mask : final mask of this generator this_sig : intermediate signal of this generator before logic operation other_masks : intermediate mask result of this generator before logic operation sig_another : signal selected other generators  0x0 <b>DIS</b> — no logic operation 0x1 <b>AND</b> — sigout = sig_another AND this_sig 0x2 <b>OR</b> — sigout = sig_another OR this_sig 0x3 <b>MASK</b> — mask = sig_another AND other_masks

## 22.4.56 Panel Interface Signal Generator Register 1\_0 (EPDC\_PIGEON\_1\_0)

parameters for timing signal generation

Address: 20F\_4000h base + 840h offset = 20F\_4840h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

## EPDC\_PIGEON\_1\_0 field descriptions

Field	Description
31–24 STATE_MASK	state_mask = (FSIFBIFDIFE) and (LSILBILDILE) , select any combination of scan states as reference point for local counter to start ticking  0x1 <b>FS</b> — FRAME SYNC 0x2 <b>FB</b> — FRAME BEGIN 0x4 <b>FD</b> — FRAME DATA 0x8 <b>FE</b> — FRAME END 0x10 <b>LS</b> — LINE SYNC 0x20 <b>LB</b> — LINE BEGIN 0x40 <b>LD</b> — LINE DATA 0x80 <b>LE</b> — LINE END
23–12 MASK_CNT	When the global counter selected through MASK_CNT_SEL matches value in this reg, pigeon local counter start ticking. 0=disable
11–8 MASK_CNT_SEL	select global counters as mask condition, use together with MASK_CNT  0x0 <b>HSTATE_CNT</b> — pclk counter within one hscan state 0x1 <b>HSTATE_CYCLE</b> — pclk cycle within one hscan state 0x2 <b>VSTATE_CNT</b> — line counter within one vscan state 0x3 <b>VSTATE_CYCLE</b> — line cycle within one vscan state 0x4 <b>FRAME_CNT</b> — frame counter 0x5 <b>FRAME_CYCLE</b> — frame cycle 0x6 <b>HCNT</b> — horizontal counter (pclk counter within one line ) 0x7 <b>VCNT</b> — vertical counter (line counter within one frame)
7–4 OFFSET	offset on pclk unit. 0=align with data, positive value means delay, minus value mean ahead. Supported range depends on panel mode
3–2 INC_SEL	event to increment local counter  0x0 <b>PCLK</b> — pclk 0x1 <b>LINE</b> — line start pulse 0x2 <b>FRAME</b> — frame start pulse 0x3 <b>SIG_ANOTHER</b> — use another signal as tick event
1 POL	polarity of signal output  0x0 <b>ACTIVE_HIGH</b> — normal signal (active high) 0x1 <b>ACTIVE_LOW</b> — inverted signal (active low)
0 EN	enable pigeon mode on this signal

## 22.4.57 Panel Interface Signal Generator Register 1\_1 (EPDC\_PIGEON\_1\_1)

parameters for timing signal generation



Address: 20F\_4000h base + 850h offset = 20F\_4850h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### EPDC\_PIGEON\_1\_1 field descriptions

Field	Description
31–16 CLR_CNT	deassert signal output when counter match this value  0x0 <b>CLEAR_USING_MASK</b> — keep active until mask off
15–0 SET_CNT	assert signal output when counter match this value  0x0 <b>START_ACTIVE</b> — start as active

## 22.4.58 Panel Interface Signal Generator Register 1\_1 (EPDC\_PIGEON\_1\_2)

parameters for timing signal generation

Address: 20F\_4000h base + 860h offset = 20F\_4860h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### EPDC\_PIGEON\_1\_2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved.
8–4 SIG_ANOTHER	select another signal for logic operation or as mask or counter tick event
3–0 SIG_LOGIC	logic operation with another signal sigout : final output signal of this generator mask : final mask of this generator this_sig : intermediate signal of this generator before logic operation other_masks : intermediate mask result of this generator before logic operation sig_another : signal selected other generators  0x0 <b>DIS</b> — no logic operation 0x1 <b>AND</b> — sigout = sig_another AND this_sig 0x2 <b>OR</b> — sigout = sig_another OR this_sig 0x3 <b>MASK</b> — mask = sig_another AND other_masks

## 22.4.59 Panel Interface Signal Generator Register 2\_0 (EPDC\_PIGEON\_2\_0)

parameters for timing signal generation

Address: 20F\_4000h base + 880h offset = 20F\_4880h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATE_MASK								MASK_CNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MASK_CNT				MASK_CNT_SEL				OFFSET				INC_SEL		POL	EN
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

### EPDC\_PIGEON\_2\_0 field descriptions

Field	Description
31–24 STATE_MASK	state_mask = (FSIFBIFDIFE) and (LSILBILDILE) , select any combination of scan states as reference point for local counter to start ticking  0x1 <b>FS</b> — FRAME SYNC 0x2 <b>FB</b> — FRAME BEGIN 0x4 <b>FD</b> — FRAME DATA 0x8 <b>FE</b> — FRAME END 0x10 <b>LS</b> — LINE SYNC 0x20 <b>LB</b> — LINE BEGIN 0x40 <b>LD</b> — LINE DATA 0x80 <b>LE</b> — LINE END
23–12 MASK_CNT	When the global counter selected through MASK_CNT_SEL matches value in this reg, pigeon local counter start ticking. 0=disable
11–8 MASK_CNT_SEL	select global counters as mask condition, use together with MASK_CNT  0x0 <b>HSTATE_CNT</b> — pclk counter within one hscan state 0x1 <b>HSTATE_CYCLE</b> — pclk cycle within one hscan state 0x2 <b>VSTATE_CNT</b> — line counter within one vscan state 0x3 <b>VSTATE_CYCLE</b> — line cycle within one vscan state 0x4 <b>FRAME_CNT</b> — frame counter 0x5 <b>FRAME_CYCLE</b> — frame cycle 0x6 <b>HCNT</b> — horizontal counter (pclk counter within one line ) 0x7 <b>VCNT</b> — vertical counter (line counter within one frame)
7–4 OFFSET	offset on pclk unit. 0=align with data, positive value means delay, minus value mean ahead. Supported range depends on panel mode
3–2 INC_SEL	event to incrmnt local counter  0x0 <b>PCLK</b> — pclk 0x1 <b>LINE</b> — line start pulse

*Table continues on the next page...*

**EPDC\_PIGEON\_2\_0 field descriptions (continued)**

Field	Description
0x2 <b>FRAME</b> — frame start pulse	
0x3 <b>SIG_ANOTHER</b> — use another signal as tick event	
1 POL	polarity of signal output  0x0 <b>ACTIVE_HIGH</b> — normal signal (active high) 0x1 <b>ACTIVE_LOW</b> — inverted signal (active low)
0 EN	enable pigeon mode on this signal

**22.4.60 Panel Interface Signal Generator Register 2\_1 (EPDC\_PIGEON\_2\_1)**

parameters for timing signal generation

Address: 20F\_4000h base + 890h offset = 20F\_4890h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLR_CNT																SET_CNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**EPDC\_PIGEON\_2\_1 field descriptions**

Field	Description
31–16 CLR_CNT	deassert signal output when counter match this value  0x0 <b>CLEAR_USING_MASK</b> — keep active until mask off
15–0 SET_CNT	assert signal output when counter match this value  0x0 <b>START_ACTIVE</b> — start as active

**22.4.61 Panel Interface Signal Generator Register 2\_1 (EPDC\_PIGEON\_2\_2)**

parameters for timing signal generation

Address: 20F\_4000h base + 8A0h offset = 20F\_48A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																SIG_ANOTHER						SIG_LOGIC									
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

## EPDC\_PIGEON\_2\_2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved.
8–4 SIG_ANOTHER	select another signal for logic operation or as mask or counter tick event
3–0 SIG_LOGIC	logic operation with another signal sigout : final output signal of this generator mask : final mask of this generator this_sig : intermediate signal of this generator before logic operation other_masks : intermediate mask result of this generator before logic operation sig_another : signal selected other generators  0x0 <b>DIS</b> — no logic operation 0x1 <b>AND</b> — sigout = sig_another AND this_sig 0x2 <b>OR</b> — sigout = sig_another OR this_sig 0x3 <b>MASK</b> — mask = sig_another AND other_masks

## 22.4.62 Panel Interface Signal Generator Register 3\_0 (EPDC\_PIGEON\_3\_0)

parameters for timing signal generation

Address: 20F\_4000h base + 8C0h offset = 20F\_48C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATE_MASK								MASK_CNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MASK_CNT				MASK_CNT_SEL				OFFSET				INC_SEL		POL	EN
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

## EPDC\_PIGEON\_3\_0 field descriptions

Field	Description
31–24 STATE_MASK	state_mask = (FSIFBIFDIFE) and (LSILBILDILE) , select any combination of scan states as reference point for local counter to start ticking  0x1 <b>FS</b> — FRAME SYNC 0x2 <b>FB</b> — FRAME BEGIN 0x4 <b>FD</b> — FRAME DATA 0x8 <b>FE</b> — FRAME END 0x10 <b>LS</b> — LINE SYNC 0x20 <b>LB</b> — LINE BEGIN

Table continues on the next page...

**EPDC\_PIGEON\_3\_0 field descriptions (continued)**

Field	Description
	0x40 <b>LD</b> — LINE DATA 0x80 <b>LE</b> — LINE END
23–12 MASK_CNT	When the global counter selected through MASK_CNT_SEL matches value in this reg, pigeon local counter start ticking. 0=disable
11–8 MASK_CNT_SEL	select global counters as mask condition, use together with MASK_CNT 0x0 <b>HSTATE_CNT</b> — pclk counter within one hscan state 0x1 <b>HSTATE_CYCLE</b> — pclk cycle within one hscan state 0x2 <b>VSTATE_CNT</b> — line counter within one vscan state 0x3 <b>VSTATE_CYCLE</b> — line cycle within one vscan state 0x4 <b>FRAME_CNT</b> — frame counter 0x5 <b>FRAME_CYCLE</b> — frame cycle 0x6 <b>HCNT</b> — horizontal counter (pclk counter within one line ) 0x7 <b>VCNT</b> — vertical counter (line counter within one frame)
7–4 OFFSET	offset on pclk unit. 0=aligne with data, positive value means delay, minus value mean ahead. Supported range depends on panel mode
3–2 INC_SEL	event to incrmnt local counter 0x0 <b>PCLK</b> — pclk 0x1 <b>LINE</b> — line start pulse 0x2 <b>FRAME</b> — frame start pulse 0x3 <b>SIG_ANOTHER</b> — use another signal as tick event
1 POL	polarity of signal output 0x0 <b>ACTIVE_HIGH</b> — normal signal (active high) 0x1 <b>ACTIVE_LOW</b> — inverted signal (active low)
0 EN	enable pigeon mode on this signal

**22.4.63 Panel Interface Signal Generator Register 3\_1 (EPDC\_PIGEON\_3\_1)**

parameters for timing signal generation

Address: 20F\_4000h base + 8D0h offset = 20F\_48D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLR_CNT																SET_CNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**EPDC\_PIGEON\_3\_1 field descriptions**

Field	Description
31–16 CLR_CNT	deassert signal output when counter match this value

*Table continues on the next page...*

**EPDC\_PIGEON\_3\_1 field descriptions (continued)**

Field	Description
	0x0 <b>CLEAR_USING_MASK</b> — keep active until mask off
15–0 SET_CNT	assert signal output when counter match this value
	0x0 <b>START_ACTIVE</b> — start as active

**22.4.64 Panel Interface Signal Generator Register 3\_1 (EPDC\_PIGEON\_3\_2)**

parameters for timing signal generation

Address: 20F\_4000h base + 8E0h offset = 20F\_48E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																									SIG_ANOTHER				SIG_LOGIC			
W	Reserved																								SIG_ANOTHER				SIG_LOGIC			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EPDC\_PIGEON\_3\_2 field descriptions**

Field	Description
31–9 -	This field is reserved. Reserved.
8–4 SIG_ANOTHER	select another signal for logic operation or as mask or counter tick event
3–0 SIG_LOGIC	logic operation with another signal sigout : final output signal of this generator mask : final mask of this generator this_sig : intermediate signal of this generator before logic operation other_masks : intermediate mask result of this generator before logic operation sig_another : signal selected other generators  0x0 <b>DIS</b> — no logic operation 0x1 <b>AND</b> — sigout = sig_another AND this_sig 0x2 <b>OR</b> — sigout = sig_another OR this_sig 0x3 <b>MASK</b> — mask = sig_another AND other_masks

**22.4.65 Panel Interface Signal Generator Register 4\_0 (EPDC\_PIGEON\_4\_0)**

parameters for timing signal generation

Address: 20F\_4000h base + 900h offset = 20F\_4900h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATE_MASK								MASK_CNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MASK_CNT				MASK_CNT_SEL				OFFSET				INC_SEL		POL	EN
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

**EPDC\_PIGEON\_4\_0 field descriptions**

Field	Description
31–24 STATE_MASK	state_mask = (FSIFBIFDIFE) and (LSILBILDILE) , select any combination of scan states as reference point for local counter to start ticking  0x1 <b>FS</b> — FRAME SYNC 0x2 <b>FB</b> — FRAME BEGIN 0x4 <b>FD</b> — FRAME DATA 0x8 <b>FE</b> — FRAME END 0x10 <b>LS</b> — LINE SYNC 0x20 <b>LB</b> — LINE BEGIN 0x40 <b>LD</b> — LINE DATA 0x80 <b>LE</b> — LINE END
23–12 MASK_CNT	When the global counter selected through MASK_CNT_SEL matches value in this reg, pigeon local counter start ticking. 0=disable
11–8 MASK_CNT_SEL	select global counters as mask condition, use together with MASK_CNT  0x0 <b>HSTATE_CNT</b> — pclk counter within one hscan state 0x1 <b>HSTATE_CYCLE</b> — pclk cycle within one hscan state 0x2 <b>VSTATE_CNT</b> — line counter within one vscan state 0x3 <b>VSTATE_CYCLE</b> — line cycle within one vscan state 0x4 <b>FRAME_CNT</b> — frame counter 0x5 <b>FRAME_CYCLE</b> — frame cycle 0x6 <b>HCNT</b> — horizontal counter (pclk counter within one line ) 0x7 <b>VCNT</b> — vertical counter (line counter within one frame)
7–4 OFFSET	offset on pclk unit. 0=align with data, positive value means delay, minus value mean ahead. Supported range depends on panel mode
3–2 INC_SEL	event to incrmnt local counter  0x0 <b>PCLK</b> — pclk 0x1 <b>LINE</b> — line start pulse 0x2 <b>FRAME</b> — frame start pulse 0x3 <b>SIG_ANOTHER</b> — use another signal as tick event
1 POL	polarity of signal output  0x0 <b>ACTIVE_HIGH</b> — normal signal (active high) 0x1 <b>ACTIVE_LOW</b> — inverted signal (active low)
0 EN	enable pigeon mode on this signal

## 22.4.66 Panel Interface Signal Generator Register 4\_1 (EPDC\_PIGEON\_4\_1)

parameters for timing signal generation

Address: 20F\_4000h base + 910h offset = 20F\_4910h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLR_CNT																SET_CNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_PIGEON\_4\_1 field descriptions

Field	Description
31–16 CLR_CNT	deassert signal output when counter match this value  0x0 <b>CLEAR_USING_MASK</b> — keep active until mask off
15–0 SET_CNT	assert signal output when counter match this value  0x0 <b>START_ACTIVE</b> — start as active

## 22.4.67 Panel Interface Signal Generator Register 4\_1 (EPDC\_PIGEON\_4\_2)

parameters for timing signal generation

Address: 20F\_4000h base + 920h offset = 20F\_4920h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																SIG_ANOTHER								SIG_LOGIC							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_PIGEON\_4\_2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved.
8–4 SIG_ANOTHER	select another signal for logic operation or as mask or counter tick event
3–0 SIG_LOGIC	logic operation with another signal sigout : final output signal of this generator mask : final mask of this generator this_sig : intermediate signal of this generator before logic operation

*Table continues on the next page...*



**EPDC\_PIGEON\_4\_2 field descriptions (continued)**

Field	Description
	other_masks : intermediate mask result of this generator before logic operation
	sig_another : signal selected other generators
0x0	<b>DIS</b> — no logic operation
0x1	<b>AND</b> — sigout = sig_another AND this_sig
0x2	<b>OR</b> — sigout = sig_another OR this_sig
0x3	<b>MASK</b> — mask = sig_another AND other_masks

## 22.4.68 Panel Interface Signal Generator Register 5\_0 (EPDC\_PIGEON\_5\_0)

parameters for timing signal generation

Address: 20F\_4000h base + 940h offset = 20F\_4940h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATE_MASK								MASK_CNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MASK_CNT				MASK_CNT_SEL				OFFSET				INC_SEL		POL	EN
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

**EPDC\_PIGEON\_5\_0 field descriptions**

Field	Description
31–24 STATE_MASK	state_mask = (FSIFBIFDIFE) and (LSILBILDILE) , select any combination of scan states as reference point for local counter to start ticking  0x1 <b>FS</b> — FRAME SYNC 0x2 <b>FB</b> — FRAME BEGIN 0x4 <b>FD</b> — FRAME DATA 0x8 <b>FE</b> — FRAME END 0x10 <b>LS</b> — LINE SYNC 0x20 <b>LB</b> — LINE BEGIN 0x40 <b>LD</b> — LINE DATA 0x80 <b>LE</b> — LINE END
23–12 MASK_CNT	When the global counter selected through MASK_CNT_SEL matches value in this reg, pigeon local counter start ticking. 0=disable
11–8 MASK_CNT_SEL	select global counters as mask condition, use together with MASK_CNT  0x0 <b>HSTATE_CNT</b> — pclk counter within one hscan state 0x1 <b>HSTATE_CYCLE</b> — pclk cycle within one hscan state 0x2 <b>VSTATE_CNT</b> — line counter within one vscan state 0x3 <b>VSTATE_CYCLE</b> — line cycle within one vscan state

*Table continues on the next page...*

**EPDC\_PIGEON\_5\_0 field descriptions (continued)**

Field	Description
	0x4 <b>FRAME_CNT</b> — frame counter 0x5 <b>FRAME_CYCLE</b> — frame cycle 0x6 <b>HCNT</b> — horizontal counter (pclk counter within one line ) 0x7 <b>VCNT</b> — vertical counter (line counter within one frame)
7–4 OFFSET	offset on pclk unit. 0=align with data, positive value means delay, minus value mean ahead. Supported range depends on panel mode
3–2 INC_SEL	event to increment local counter  0x0 <b>PCLK</b> — pclk 0x1 <b>LINE</b> — line start pulse 0x2 <b>FRAME</b> — frame start pulse 0x3 <b>SIG_ANOTHER</b> — use another signal as tick event
1 POL	polarity of signal output  0x0 <b>ACTIVE_HIGH</b> — normal signal (active high) 0x1 <b>ACTIVE_LOW</b> — inverted signal (active low)
0 EN	enable pigeon mode on this signal

## 22.4.69 Panel Interface Signal Generator Register 5\_1 (EPDC\_PIGEON\_5\_1)

parameters for timing signal generation

Address: 20F\_4000h base + 950h offset = 20F\_4950h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLR_CNT																SET_CNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**EPDC\_PIGEON\_5\_1 field descriptions**

Field	Description
31–16 CLR_CNT	deassert signal output when counter match this value  0x0 <b>CLEAR_USING_MASK</b> — keep active until mask off
15–0 SET_CNT	assert signal output when counter match this value  0x0 <b>START_ACTIVE</b> — start as active

## 22.4.70 Panel Interface Signal Generator Register 5\_1 (EPDC\_PIGEON\_5\_2)

parameters for timing signal generation

Address: 20F\_4000h base + 960h offset = 20F\_4960h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																SIG_ANOTHER						SIG_LOGIC									
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_PIGEON\_5\_2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved.
8–4 SIG_ANOTHER	select another signal for logic operation or as mask or counter tick event
3–0 SIG_LOGIC	<p>logic operation with another signal</p> <p>sigout : final output signal of this generator</p> <p>mask : final mask of this generator</p> <p>this_sig : intermediate signal of this generator before logic operation</p> <p>other_masks : intermediate mask result of this generator before logic operation</p> <p>sig_another : signal selected other generators</p> <p>0x0 <b>DIS</b> — no logic operation</p> <p>0x1 <b>AND</b> — sigout = sig_another AND this_sig</p> <p>0x2 <b>OR</b> — sigout = sig_another OR this_sig</p> <p>0x3 <b>MASK</b> — mask = sig_another AND other_masks</p>

## 22.4.71 Panel Interface Signal Generator Register 6\_0 (EPDC\_PIGEON\_6\_0)

parameters for timing signal generation

Address: 20F\_4000h base + 980h offset = 20F\_4980h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

## EPDC\_PIGEON\_6\_0 field descriptions

Field	Description
31–24 STATE_MASK	state_mask = (FSIFBIFDIFE) and (LSILBILDILE) , select any combination of scan states as reference point for local counter to start ticking  0x1 <b>FS</b> — FRAME SYNC 0x2 <b>FB</b> — FRAME BEGIN 0x4 <b>FD</b> — FRAME DATA 0x8 <b>FE</b> — FRAME END 0x10 <b>LS</b> — LINE SYNC 0x20 <b>LB</b> — LINE BEGIN 0x40 <b>LD</b> — LINE DATA 0x80 <b>LE</b> — LINE END
23–12 MASK_CNT	When the global counter selected through MASK_CNT_SEL matches value in this reg, pigeon local counter start ticking. 0=disable
11–8 MASK_CNT_SEL	select global counters as mask condition, use together with MASK_CNT  0x0 <b>HSTATE_CNT</b> — pclk counter within one hscan state 0x1 <b>HSTATE_CYCLE</b> — pclk cycle within one hscan state 0x2 <b>VSTATE_CNT</b> — line counter within one vscan state 0x3 <b>VSTATE_CYCLE</b> — line cycle within one vscan state 0x4 <b>FRAME_CNT</b> — frame counter 0x5 <b>FRAME_CYCLE</b> — frame cycle 0x6 <b>HCNT</b> — horizontal counter (pclk counter within one line ) 0x7 <b>VCNT</b> — vertical counter (line counter within one frame)
7–4 OFFSET	offset on pclk unit. 0=align with data, positive value means delay, minus value mean ahead. Supported range depends on panel mode
3–2 INC_SEL	event to increment local counter  0x0 <b>PCLK</b> — pclk 0x1 <b>LINE</b> — line start pulse 0x2 <b>FRAME</b> — frame start pulse 0x3 <b>SIG_ANOTHER</b> — use another signal as tick event
1 POL	polarity of signal output  0x0 <b>ACTIVE_HIGH</b> — normal signal (active high) 0x1 <b>ACTIVE_LOW</b> — inverted signal (active low)
0 EN	enable pigeon mode on this signal

## 22.4.72 Panel Interface Signal Generator Register 6\_1 (EPDC\_PIGEON\_6\_1)

parameters for timing signal generation

Address: 20F\_4000h base + 990h offset = 20F\_4990h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLR_CNT																SET_CNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_PIGEON\_6\_1 field descriptions

Field	Description
31–16 CLR_CNT	deassert signal output when counter match this value  0x0 <b>CLEAR_USING_MASK</b> — keep active until mask off
15–0 SET_CNT	assert signal output when counter match this value  0x0 <b>START_ACTIVE</b> — start as active

## 22.4.73 Panel Interface Signal Generator Register 6\_1 (EPDC\_PIGEON\_6\_2)

parameters for timing signal generation

Address: 20F\_4000h base + 9A0h offset = 20F\_49A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																SIG_ANOTHER								SIG_LOGIC							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_PIGEON\_6\_2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved.
8–4 SIG_ANOTHER	select another signal for logic operation or as mask or counter tick event
3–0 SIG_LOGIC	logic operation with another signal sigout : final output signal of this generator mask : final mask of this generator this_sig : intermediate signal of this generator before logic operation other_masks : intermediate mask result of this generator before logic operation sig_another : signal selected other generators  0x0 <b>DIS</b> — no logic operation 0x1 <b>AND</b> — sigout = sig_another AND this_sig 0x2 <b>OR</b> — sigout = sig_another OR this_sig 0x3 <b>MASK</b> — mask = sig_another AND other_masks

## 22.4.74 Panel Interface Signal Generator Register 7\_0 (EPDC\_PIGEON\_7\_0)

parameters for timing signal generation

Address: 20F\_4000h base + 9C0h offset = 20F\_49C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATE_MASK								MASK_CNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MASK_CNT				MASK_CNT_SEL				OFFSET				INC_SEL		POL	EN
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

### EPDC\_PIGEON\_7\_0 field descriptions

Field	Description
31–24 STATE_MASK	state_mask = (FSIFBIFDIFE) and (LSILBILDILE) , select any combination of scan states as reference point for local counter to start ticking  0x1 <b>FS</b> — FRAME SYNC 0x2 <b>FB</b> — FRAME BEGIN 0x4 <b>FD</b> — FRAME DATA 0x8 <b>FE</b> — FRAME END 0x10 <b>LS</b> — LINE SYNC 0x20 <b>LB</b> — LINE BEGIN 0x40 <b>LD</b> — LINE DATA 0x80 <b>LE</b> — LINE END
23–12 MASK_CNT	When the global counter selected through MASK_CNT_SEL matches value in this reg, pigeon local counter start ticking. 0=disable
11–8 MASK_CNT_SEL	select global counters as mask condition, use together with MASK_CNT  0x0 <b>HSTATE_CNT</b> — pclk counter within one hscan state 0x1 <b>HSTATE_CYCLE</b> — pclk cycle within one hscan state 0x2 <b>VSTATE_CNT</b> — line counter within one vscan state 0x3 <b>VSTATE_CYCLE</b> — line cycle within one vscan state 0x4 <b>FRAME_CNT</b> — frame counter 0x5 <b>FRAME_CYCLE</b> — frame cycle 0x6 <b>HCNT</b> — horizontal counter (pclk counter within one line ) 0x7 <b>VCNT</b> — vertical counter (line counter within one frame)
7–4 OFFSET	offset on pclk unit. 0=align with data, positive value means delay, minus value mean ahead. Supported range depends on panel mode
3–2 INC_SEL	event to incrmnt local counter  0x0 <b>PCLK</b> — pclk 0x1 <b>LINE</b> — line start pulse

*Table continues on the next page...*

**EPDC\_PIGEON\_7\_0 field descriptions (continued)**

Field	Description
	0x2 <b>FRAME</b> — frame start pulse 0x3 <b>SIG_ANOTHER</b> — use another signal as tick event
1 POL	polarity of signal output  0x0 <b>ACTIVE_HIGH</b> — normal signal (active high) 0x1 <b>ACTIVE_LOW</b> — inverted signal (active low)
0 EN	enable pigeon mode on this signal

## 22.4.75 Panel Interface Signal Generator Register 7\_1 (EPDC\_PIGEON\_7\_1)

parameters for timing signal generation

Address: 20F\_4000h base + 9D0h offset = 20F\_49D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLR_CNT																SET_CNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**EPDC\_PIGEON\_7\_1 field descriptions**

Field	Description
31–16 CLR_CNT	deassert signal output when counter match this value  0x0 <b>CLEAR_USING_MASK</b> — keep active until mask off
15–0 SET_CNT	assert signal output when counter match this value  0x0 <b>START_ACTIVE</b> — start as active

## 22.4.76 Panel Interface Signal Generator Register 7\_1 (EPDC\_PIGEON\_7\_2)

parameters for timing signal generation

Address: 20F\_4000h base + 9E0h offset = 20F\_49E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																SIG_ANOTHER						SIG_LOGIC									
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

## EPDC\_PIGEON\_7\_2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved.
8–4 SIG_ANOTHER	select another signal for logic operation or as mask or counter tick event
3–0 SIG_LOGIC	logic operation with another signal sigout : final output signal of this generator mask : final mask of this generator this_sig : intermediate signal of this generator before logic operation other_masks : intermediate mask result of this generator before logic operation sig_another : signal selected other generators  0x0 <b>DIS</b> — no logic operation 0x1 <b>AND</b> — sigout = sig_another AND this_sig 0x2 <b>OR</b> — sigout = sig_another OR this_sig 0x3 <b>MASK</b> — mask = sig_another AND other_masks

## 22.4.77 Panel Interface Signal Generator Register 8\_0 (EPDC\_PIGEON\_8\_0)

parameters for timing signal generation

Address: 20F\_4000h base + A00h offset = 20F\_4A00h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATE_MASK								MASK_CNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MASK_CNT				MASK_CNT_SEL				OFFSET				INC_SEL		POL	EN
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

## EPDC\_PIGEON\_8\_0 field descriptions

Field	Description
31–24 STATE_MASK	state_mask = (FSIFBIFDIFE) and (LSILBILDILE) , select any combination of scan states as reference point for local counter to start ticking  0x1 <b>FS</b> — FRAME SYNC 0x2 <b>FB</b> — FRAME BEGIN 0x4 <b>FD</b> — FRAME DATA 0x8 <b>FE</b> — FRAME END 0x10 <b>LS</b> — LINE SYNC 0x20 <b>LB</b> — LINE BEGIN

Table continues on the next page...



**EPDC\_PIGEON\_8\_0 field descriptions (continued)**

Field	Description
	0x40 <b>LD</b> — LINE DATA 0x80 <b>LE</b> — LINE END
23–12 MASK_CNT	When the global counter selected through MASK_CNT_SEL matches value in this reg, pigeon local counter start ticking. 0=disable
11–8 MASK_CNT_SEL	select global counters as mask condition, use together with MASK_CNT 0x0 <b>HSTATE_CNT</b> — pclk counter within one hscan state 0x1 <b>HSTATE_CYCLE</b> — pclk cycle within one hscan state 0x2 <b>VSTATE_CNT</b> — line counter within one vscan state 0x3 <b>VSTATE_CYCLE</b> — line cycle within one vscan state 0x4 <b>FRAME_CNT</b> — frame counter 0x5 <b>FRAME_CYCLE</b> — frame cycle 0x6 <b>HCNT</b> — horizontal counter (pclk counter within one line ) 0x7 <b>VCNT</b> — vertical counter (line counter within one frame)
7–4 OFFSET	offset on pclk unit. 0=align with data, positive value means delay, minus value mean ahead. Supported range depends on panel mode
3–2 INC_SEL	event to increment local counter 0x0 <b>PCLK</b> — pclk 0x1 <b>LINE</b> — line start pulse 0x2 <b>FRAME</b> — frame start pulse 0x3 <b>SIG_ANOTHER</b> — use another signal as tick event
1 POL	polarity of signal output 0x0 <b>ACTIVE_HIGH</b> — normal signal (active high) 0x1 <b>ACTIVE_LOW</b> — inverted signal (active low)
0 EN	enable pigeon mode on this signal

**22.4.78 Panel Interface Signal Generator Register 8\_1 (EPDC\_PIGEON\_8\_1)**

parameters for timing signal generation

Address: 20F\_4000h base + A10h offset = 20F\_4A10h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLR_CNT																SET_CNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**EPDC\_PIGEON\_8\_1 field descriptions**

Field	Description
31–16 CLR_CNT	deassert signal output when counter match this value

*Table continues on the next page...*

**EPDC\_PIGEON\_8\_1 field descriptions (continued)**

Field	Description
	0x0 <b>CLEAR_USING_MASK</b> — keep active until mask off
15–0 SET_CNT	assert signal output when counter match this value
	0x0 <b>START_ACTIVE</b> — start as active

**22.4.79 Panel Interface Signal Generator Register 8\_1 (EPDC\_PIGEON\_8\_2)**

parameters for timing signal generation

Address: 20F\_4000h base + A20h offset = 20F\_4A20h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																SIG_ANOTHER				SIG_LOGIC											
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**EPDC\_PIGEON\_8\_2 field descriptions**

Field	Description
31–9 -	This field is reserved. Reserved.
8–4 SIG_ANOTHER	select another signal for logic operation or as mask or counter tick event
3–0 SIG_LOGIC	<p>logic operation with another signal</p> <p>sigout : final output signal of this generator</p> <p>mask : final mask of this generator</p> <p>this_sig : intermediate signal of this generator before logic operation</p> <p>other_masks : intermediate mask result of this generator before logic operation</p> <p>sig_another : signal selected other generators</p> <p>0x0 <b>DIS</b> — no logic operation</p> <p>0x1 <b>AND</b> — sigout = sig_another AND this_sig</p> <p>0x2 <b>OR</b> — sigout = sig_another OR this_sig</p> <p>0x3 <b>MASK</b> — mask = sig_another AND other_masks</p>

**22.4.80 Panel Interface Signal Generator Register 9\_0 (EPDC\_PIGEON\_9\_0)**

parameters for timing signal generation

Address: 20F\_4000h base + A40h offset = 20F\_4A40h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATE_MASK								MASK_CNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MASK_CNT				MASK_CNT_SEL				OFFSET				INC_SEL		POL	EN
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

**EPDC\_PIGEON\_9\_0 field descriptions**

Field	Description
31–24 STATE_MASK	state_mask = (FSIFBIFDIFE) and (LSILBILDILE) , select any combination of scan states as reference point for local counter to start ticking  0x1 <b>FS</b> — FRAME SYNC 0x2 <b>FB</b> — FRAME BEGIN 0x4 <b>FD</b> — FRAME DATA 0x8 <b>FE</b> — FRAME END 0x10 <b>LS</b> — LINE SYNC 0x20 <b>LB</b> — LINE BEGIN 0x40 <b>LD</b> — LINE DATA 0x80 <b>LE</b> — LINE END
23–12 MASK_CNT	When the global counter selected through MASK_CNT_SEL matches value in this reg, pigeon local counter start ticking. 0=disable
11–8 MASK_CNT_SEL	select global counters as mask condition, use together with MASK_CNT  0x0 <b>HSTATE_CNT</b> — pclk counter within one hscan state 0x1 <b>HSTATE_CYCLE</b> — pclk cycle within one hscan state 0x2 <b>VSTATE_CNT</b> — line counter within one vscan state 0x3 <b>VSTATE_CYCLE</b> — line cycle within one vscan state 0x4 <b>FRAME_CNT</b> — frame counter 0x5 <b>FRAME_CYCLE</b> — frame cycle 0x6 <b>HCNT</b> — horizontal counter (pclk counter within one line ) 0x7 <b>VCNT</b> — vertical counter (line counter within one frame)
7–4 OFFSET	offset on pclk unit. 0=align with data, positive value means delay, minus value mean ahead. Supported range depends on panel mode
3–2 INC_SEL	event to incrmnt local counter  0x0 <b>PCLK</b> — pclk 0x1 <b>LINE</b> — line start pulse 0x2 <b>FRAME</b> — frame start pulse 0x3 <b>SIG_ANOTHER</b> — use another signal as tick event
1 POL	polarity of signal output  0x0 <b>ACTIVE_HIGH</b> — normal signal (active high) 0x1 <b>ACTIVE_LOW</b> — inverted signal (active low)
0 EN	enable pigeon mode on this signal

## 22.4.81 Panel Interface Signal Generator Register 9\_1 (EPDC\_PIGEON\_9\_1)

parameters for timing signal generation

Address: 20F\_4000h base + A50h offset = 20F\_4A50h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLR_CNT																SET_CNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_PIGEON\_9\_1 field descriptions

Field	Description
31–16 CLR_CNT	deassert signal output when counter match this value  0x0 <b>CLEAR_USING_MASK</b> — keep active until mask off
15–0 SET_CNT	assert signal output when counter match this value  0x0 <b>START_ACTIVE</b> — start as active

## 22.4.82 Panel Interface Signal Generator Register 9\_1 (EPDC\_PIGEON\_9\_2)

parameters for timing signal generation

Address: 20F\_4000h base + A60h offset = 20F\_4A60h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																SIG_ANOTHER								SIG_LOGIC							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_PIGEON\_9\_2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved.
8–4 SIG_ANOTHER	select another signal for logic operation or as mask or counter tick event
3–0 SIG_LOGIC	logic operation with another signal sigout : final output signal of this generator mask : final mask of this generator this_sig : intermediate signal of this generator before logic operation

*Table continues on the next page...*

**EPDC\_PIGEON\_9\_2 field descriptions (continued)**

Field	Description
	other_masks : intermediate mask result of this generator before logic operation
	sig_another : signal selected other generators
0x0	<b>DIS</b> — no logic operation
0x1	<b>AND</b> — sigout = sig_another AND this_sig
0x2	<b>OR</b> — sigout = sig_another OR this_sig
0x3	<b>MASK</b> — mask = sig_another AND other_masks

## 22.4.83 Panel Interface Signal Generator Register 10\_0 (EPDC\_PIGEON\_10\_0)

parameters for timing signal generation

Address: 20F\_4000h base + A80h offset = 20F\_4A80h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATE_MASK								MASK_CNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MASK_CNT				MASK_CNT_SEL				OFFSET				INC_SEL		POL	EN
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

**EPDC\_PIGEON\_10\_0 field descriptions**

Field	Description
31–24 STATE_MASK	state_mask = (FSIFBIFDIFE) and (LSILBILDILE) , select any combination of scan states as reference point for local counter to start ticking  0x1 <b>FS</b> — FRAME SYNC 0x2 <b>FB</b> — FRAME BEGIN 0x4 <b>FD</b> — FRAME DATA 0x8 <b>FE</b> — FRAME END 0x10 <b>LS</b> — LINE SYNC 0x20 <b>LB</b> — LINE BEGIN 0x40 <b>LD</b> — LINE DATA 0x80 <b>LE</b> — LINE END
23–12 MASK_CNT	When the global counter selected through MASK_CNT_SEL matches value in this reg, pigeon local counter start ticking. 0=disable
11–8 MASK_CNT_SEL	select global counters as mask condition, use together with MASK_CNT  0x0 <b>HSTATE_CNT</b> — pclk counter within one hscan state 0x1 <b>HSTATE_CYCLE</b> — pclk cycle within one hscan state 0x2 <b>VSTATE_CNT</b> — line counter within one vscan state 0x3 <b>VSTATE_CYCLE</b> — line cycle within one vscan state

*Table continues on the next page...*

**EPDC\_PIGEON\_10\_0 field descriptions (continued)**

Field	Description
	0x4 <b>FRAME_CNT</b> — frame counter 0x5 <b>FRAME_CYCLE</b> — frame cycle 0x6 <b>HCNT</b> — horizontal counter (pclk counter within one line ) 0x7 <b>VCNT</b> — vertical counter (line counter within one frame)
7–4 OFFSET	offset on pclk unit. 0=aligns with data, positive value means delay, minus value means ahead. Supported range depends on panel mode
3–2 INC_SEL	event to increment local counter  0x0 <b>PCLK</b> — pclk 0x1 <b>LINE</b> — line start pulse 0x2 <b>FRAME</b> — frame start pulse 0x3 <b>SIG_ANOTHER</b> — use another signal as tick event
1 POL	polarity of signal output  0x0 <b>ACTIVE_HIGH</b> — normal signal (active high) 0x1 <b>ACTIVE_LOW</b> — inverted signal (active low)
0 EN	enable pigeon mode on this signal

## 22.4.84 Panel Interface Signal Generator Register 10\_1 (EPDC\_PIGEON\_10\_1)

parameters for timing signal generation

Address: 20F\_4000h base + A90h offset = 20F\_4A90h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLR_CNT																SET_CNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**EPDC\_PIGEON\_10\_1 field descriptions**

Field	Description
31–16 CLR_CNT	deassert signal output when counter match this value  0x0 <b>CLEAR_USING_MASK</b> — keep active until mask off
15–0 SET_CNT	assert signal output when counter match this value  0x0 <b>START_ACTIVE</b> — start as active

## 22.4.85 Panel Interface Signal Generator Register 10\_1 (EPDC\_PIGEON\_10\_2)

parameters for timing signal generation

Address: 20F\_4000h base + AA0h offset = 20F\_4AA0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																SIG_ANOTHER						SIG_LOGIC									
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_PIGEON\_10\_2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved.
8–4 SIG_ANOTHER	select another signal for logic operation or as mask or counter tick event
3–0 SIG_LOGIC	logic operation with another signal sigout : final output signal of this generator mask : final mask of this generator this_sig : intermediate signal of this generator before logic operation other_masks : intermediate mask result of this generator before logic operation sig_another : signal selected other generators  0x0 <b>DIS</b> — no logic operation 0x1 <b>AND</b> — sigout = sig_another AND this_sig 0x2 <b>OR</b> — sigout = sig_another OR this_sig 0x3 <b>MASK</b> — mask = sig_another AND other_masks

## 22.4.86 Panel Interface Signal Generator Register 11\_0 (EPDC\_PIGEON\_11\_0)

parameters for timing signal generation

Address: 20F\_4000h base + AC0h offset = 20F\_4AC0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

## EPDC\_PIGEON\_11\_0 field descriptions

Field	Description
31–24 STATE_MASK	state_mask = (FSIFBIFDIFE) and (LSILBILDILE) , select any combination of scan states as reference point for local counter to start ticking  0x1 <b>FS</b> — FRAME SYNC 0x2 <b>FB</b> — FRAME BEGIN 0x4 <b>FD</b> — FRAME DATA 0x8 <b>FE</b> — FRAME END 0x10 <b>LS</b> — LINE SYNC 0x20 <b>LB</b> — LINE BEGIN 0x40 <b>LD</b> — LINE DATA 0x80 <b>LE</b> — LINE END
23–12 MASK_CNT	When the global counter selected through MASK_CNT_SEL matches value in this reg, pigeon local counter start ticking. 0=disable
11–8 MASK_CNT_SEL	select global counters as mask condition, use together with MASK_CNT  0x0 <b>HSTATE_CNT</b> — pclk counter within one hscan state 0x1 <b>HSTATE_CYCLE</b> — pclk cycle within one hscan state 0x2 <b>VSTATE_CNT</b> — line counter within one vscan state 0x3 <b>VSTATE_CYCLE</b> — line cycle within one vscan state 0x4 <b>FRAME_CNT</b> — frame counter 0x5 <b>FRAME_CYCLE</b> — frame cycle 0x6 <b>HCNT</b> — horizontal counter (pclk counter within one line ) 0x7 <b>VCNT</b> — vertical counter (line counter within one frame)
7–4 OFFSET	offset on pclk unit. 0=align with data, positive value means delay, minus value mean ahead. Supported range depends on panel mode
3–2 INC_SEL	event to increment local counter  0x0 <b>PCLK</b> — pclk 0x1 <b>LINE</b> — line start pulse 0x2 <b>FRAME</b> — frame start pulse 0x3 <b>SIG_ANOTHER</b> — use another signal as tick event
1 POL	polarity of signal output  0x0 <b>ACTIVE_HIGH</b> — normal signal (active high) 0x1 <b>ACTIVE_LOW</b> — inverted signal (active low)
0 EN	enable pigeon mode on this signal

## 22.4.87 Panel Interface Signal Generator Register 11\_1 (EPDC\_PIGEON\_11\_1)

parameters for timing signal generation



Address: 20F\_4000h base + AD0h offset = 20F\_4AD0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLR_CNT																SET_CNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**EPDC\_PIGEON\_11\_1 field descriptions**

Field	Description
31–16 CLR_CNT	deassert signal output when counter match this value  0x0 <b>CLEAR_USING_MASK</b> — keep active until mask off
15–0 SET_CNT	assert signal output when counter match this value  0x0 <b>START_ACTIVE</b> — start as active

## 22.4.88 Panel Interface Signal Generator Register 11\_1 (EPDC\_PIGEON\_11\_2)

parameters for timing signal generation

Address: 20F\_4000h base + AE0h offset = 20F\_4AE0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																SIG_ANOTHER				SIG_LOGIC											
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**EPDC\_PIGEON\_11\_2 field descriptions**

Field	Description
31–9 -	This field is reserved. Reserved.
8–4 SIG_ANOTHER	select another signal for logic operation or as mask or counter tick event
3–0 SIG_LOGIC	logic operation with another signal sigout : final output signal of this generator mask : final mask of this generator this_sig : intermediate signal of this generator before logic operation other_masks : intermediate mask result of this generator before logic operation sig_another : signal selected other generators  0x0 <b>DIS</b> — no logic operation 0x1 <b>AND</b> — sigout = sig_another AND this_sig 0x2 <b>OR</b> — sigout = sig_another OR this_sig 0x3 <b>MASK</b> — mask = sig_another AND other_masks

## 22.4.89 Panel Interface Signal Generator Register 12\_0 (EPDC\_PIGEON\_12\_0)

parameters for timing signal generation

Address: 20F\_4000h base + B00h offset = 20F\_4B00h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATE_MASK								MASK_CNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MASK_CNT				MASK_CNT_SEL				OFFSET				INC_SEL		POL	EN
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

### EPDC\_PIGEON\_12\_0 field descriptions

Field	Description
31–24 STATE_MASK	state_mask = (FSIFBIFDIFE) and (LSILBILDILE) , select any combination of scan states as reference point for local counter to start ticking  0x1 <b>FS</b> — FRAME SYNC 0x2 <b>FB</b> — FRAME BEGIN 0x4 <b>FD</b> — FRAME DATA 0x8 <b>FE</b> — FRAME END 0x10 <b>LS</b> — LINE SYNC 0x20 <b>LB</b> — LINE BEGIN 0x40 <b>LD</b> — LINE DATA 0x80 <b>LE</b> — LINE END
23–12 MASK_CNT	When the global counter selected through MASK_CNT_SEL matches value in this reg, pigeon local counter start ticking. 0=disable
11–8 MASK_CNT_SEL	select global counters as mask condition, use together with MASK_CNT  0x0 <b>HSTATE_CNT</b> — pclk counter within one hscan state 0x1 <b>HSTATE_CYCLE</b> — pclk cycle within one hscan state 0x2 <b>VSTATE_CNT</b> — line counter within one vscan state 0x3 <b>VSTATE_CYCLE</b> — line cycle within one vscan state 0x4 <b>FRAME_CNT</b> — frame counter 0x5 <b>FRAME_CYCLE</b> — frame cycle 0x6 <b>HCNT</b> — horizontal counter (pclk counter within one line ) 0x7 <b>VCNT</b> — vertical counter (line counter within one frame)
7–4 OFFSET	offset on pclk unit. 0=align with data, positive value means delay, minus value mean ahead. Supported range depends on panel mode
3–2 INC_SEL	event to incrmnt local counter  0x0 <b>PCLK</b> — pclk 0x1 <b>LINE</b> — line start pulse

*Table continues on the next page...*

**EPDC\_PIGEON\_12\_0 field descriptions (continued)**

Field	Description
0x2 <b>FRAME</b>	— frame start pulse
0x3 <b>SIG_ANOTHER</b>	— use another signal as tick event
1 POL	polarity of signal output  0x0 <b>ACTIVE_HIGH</b> — normal signal (active high) 0x1 <b>ACTIVE_LOW</b> — inverted signal (active low)
0 EN	enable pigeon mode on this signal

## 22.4.90 Panel Interface Signal Generator Register 12\_1 (EPDC\_PIGEON\_12\_1)

parameters for timing signal generation

Address: 20F\_4000h base + B10h offset = 20F\_4B10h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLR_CNT																SET_CNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**EPDC\_PIGEON\_12\_1 field descriptions**

Field	Description
31–16 CLR_CNT	deassert signal output when counter match this value  0x0 <b>CLEAR_USING_MASK</b> — keep active until mask off
15–0 SET_CNT	assert signal output when counter match this value  0x0 <b>START_ACTIVE</b> — start as active

## 22.4.91 Panel Interface Signal Generator Register 12\_1 (EPDC\_PIGEON\_12\_2)

parameters for timing signal generation

Address: 20F\_4000h base + B20h offset = 20F\_4B20h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																SIG_ANOTHER						SIG_LOGIC									
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

## EPDC\_PIGEON\_12\_2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved.
8–4 SIG_ANOTHER	select another signal for logic operation or as mask or counter tick event
3–0 SIG_LOGIC	logic operation with another signal sigout : final output signal of this generator mask : final mask of this generator this_sig : intermediate signal of this generator before logic operation other_masks : intermediate mask result of this generator before logic operation sig_another : signal selected other generators  0x0 <b>DIS</b> — no logic operation 0x1 <b>AND</b> — sigout = sig_another AND this_sig 0x2 <b>OR</b> — sigout = sig_another OR this_sig 0x3 <b>MASK</b> — mask = sig_another AND other_masks

## 22.4.92 Panel Interface Signal Generator Register 13\_0 (EPDC\_PIGEON\_13\_0)

parameters for timing signal generation

Address: 20F\_4000h base + B40h offset = 20F\_4B40h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATE_MASK								MASK_CNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MASK_CNT				MASK_CNT_SEL				OFFSET				INC_SEL		POL	EN
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

## EPDC\_PIGEON\_13\_0 field descriptions

Field	Description
31–24 STATE_MASK	state_mask = (FSIFBIFDIFE) and (LSILBILDILE) , select any combination of scan states as reference point for local counter to start ticking  0x1 <b>FS</b> — FRAME SYNC 0x2 <b>FB</b> — FRAME BEGIN 0x4 <b>FD</b> — FRAME DATA 0x8 <b>FE</b> — FRAME END 0x10 <b>LS</b> — LINE SYNC 0x20 <b>LB</b> — LINE BEGIN

Table continues on the next page...

**EPDC\_PIGEON\_13\_0 field descriptions (continued)**

Field	Description
	0x40 <b>LD</b> — LINE DATA 0x80 <b>LE</b> — LINE END
23–12 MASK_CNT	When the global counter selected through MASK_CNT_SEL matches value in this reg, pigeon local counter start ticking. 0=disable
11–8 MASK_CNT_SEL	select global counters as mask condition, use together with MASK_CNT 0x0 <b>HSTATE_CNT</b> — pclk counter within one hscan state 0x1 <b>HSTATE_CYCLE</b> — pclk cycle within one hscan state 0x2 <b>VSTATE_CNT</b> — line counter within one vscan state 0x3 <b>VSTATE_CYCLE</b> — line cycle within one vscan state 0x4 <b>FRAME_CNT</b> — frame counter 0x5 <b>FRAME_CYCLE</b> — frame cycle 0x6 <b>HCNT</b> — horizontal counter (pclk counter within one line ) 0x7 <b>VCNT</b> — vertical counter (line counter within one frame)
7–4 OFFSET	offset on pclk unit. 0=align with data, positive value means delay, minus value mean ahead. Supported range depends on panel mode
3–2 INC_SEL	event to incrmnt local counter 0x0 <b>PCLK</b> — pclk 0x1 <b>LINE</b> — line start pulse 0x2 <b>FRAME</b> — frame start pulse 0x3 <b>SIG_ANOTHER</b> — use another signal as tick event
1 POL	polarity of signal output 0x0 <b>ACTIVE_HIGH</b> — normal signal (active high) 0x1 <b>ACTIVE_LOW</b> — inverted signal (active low)
0 EN	enable pigeon mode on this signal

**22.4.93 Panel Interface Signal Generator Register 13\_1 (EPDC\_PIGEON\_13\_1)**

parameters for timing signal generation

Address: 20F\_4000h base + B50h offset = 20F\_4B50h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLR_CNT																SET_CNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**EPDC\_PIGEON\_13\_1 field descriptions**

Field	Description
31–16 CLR_CNT	deassert signal output when counter match this value

*Table continues on the next page...*

**EPDC\_PIGEON\_13\_1 field descriptions (continued)**

Field	Description
	0x0 <b>CLEAR_USING_MASK</b> — keep active until mask off
15–0 SET_CNT	assert signal output when counter match this value
	0x0 <b>START_ACTIVE</b> — start as active

**22.4.94 Panel Interface Signal Generator Register 13\_1 (EPDC\_PIGEON\_13\_2)**

parameters for timing signal generation

Address: 20F\_4000h base + B60h offset = 20F\_4B60h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																									SIG_ANOTHER				SIG_LOGIC			
W	Reserved																								SIG_ANOTHER				SIG_LOGIC			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EPDC\_PIGEON\_13\_2 field descriptions**

Field	Description
31–9 -	This field is reserved. Reserved.
8–4 SIG_ANOTHER	select another signal for logic operation or as mask or counter tick event
3–0 SIG_LOGIC	logic operation with another signal sigout : final output signal of this generator mask : final mask of this generator this_sig : intermediate signal of this generator before logic operation other_masks : intermediate mask result of this generator before logic operation sig_another : signal selected other generators  0x0 <b>DIS</b> — no logic operation 0x1 <b>AND</b> — sigout = sig_another AND this_sig 0x2 <b>OR</b> — sigout = sig_another OR this_sig 0x3 <b>MASK</b> — mask = sig_another AND other_masks

**22.4.95 Panel Interface Signal Generator Register 14\_0 (EPDC\_PIGEON\_14\_0)**

parameters for timing signal generation

Address: 20F\_4000h base + B80h offset = 20F\_4B80h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATE_MASK								MASK_CNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MASK_CNT				MASK_CNT_SEL				OFFSET				INC_SEL		POL	EN
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

## EPDC\_PIGEON\_14\_0 field descriptions

Field	Description
31–24 STATE_MASK	state_mask = (FSIFBIFDIFE) and (LSILBILDILE) , select any combination of scan states as reference point for local counter to start ticking  0x1 <b>FS</b> — FRAME SYNC 0x2 <b>FB</b> — FRAME BEGIN 0x4 <b>FD</b> — FRAME DATA 0x8 <b>FE</b> — FRAME END 0x10 <b>LS</b> — LINE SYNC 0x20 <b>LB</b> — LINE BEGIN 0x40 <b>LD</b> — LINE DATA 0x80 <b>LE</b> — LINE END
23–12 MASK_CNT	When the global counter selected through MASK_CNT_SEL matches value in this reg, pigeon local counter start ticking. 0=disable
11–8 MASK_CNT_SEL	select global counters as mask condition, use together with MASK_CNT  0x0 <b>HSTATE_CNT</b> — pclk counter within one hscan state 0x1 <b>HSTATE_CYCLE</b> — pclk cycle within one hscan state 0x2 <b>VSTATE_CNT</b> — line counter within one vscan state 0x3 <b>VSTATE_CYCLE</b> — line cycle within one vscan state 0x4 <b>FRAME_CNT</b> — frame counter 0x5 <b>FRAME_CYCLE</b> — frame cycle 0x6 <b>HCNT</b> — horizontal counter (pclk counter within one line ) 0x7 <b>VCNT</b> — vertical counter (line counter within one frame)
7–4 OFFSET	offset on pclk unit. 0=align with data, positive value means delay, minus value mean ahead. Supported range depends on panel mode
3–2 INC_SEL	event to incrmnt local counter  0x0 <b>PCLK</b> — pclk 0x1 <b>LINE</b> — line start pulse 0x2 <b>FRAME</b> — frame start pulse 0x3 <b>SIG_ANOTHER</b> — use another signal as tick event
1 POL	polarity of signal output  0x0 <b>ACTIVE_HIGH</b> — normal signal (active high) 0x1 <b>ACTIVE_LOW</b> — inverted signal (active low)
0 EN	enable pigeon mode on this signal

## 22.4.96 Panel Interface Signal Generator Register 14\_1 (EPDC\_PIGEON\_14\_1)

parameters for timing signal generation

Address: 20F\_4000h base + B90h offset = 20F\_4B90h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLR_CNT																SET_CNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_PIGEON\_14\_1 field descriptions

Field	Description
31–16 CLR_CNT	deassert signal output when counter match this value  0x0 <b>CLEAR_USING_MASK</b> — keep active until mask off
15–0 SET_CNT	assert signal output when counter match this value  0x0 <b>START_ACTIVE</b> — start as active

## 22.4.97 Panel Interface Signal Generator Register 14\_1 (EPDC\_PIGEON\_14\_2)

parameters for timing signal generation

Address: 20F\_4000h base + BA0h offset = 20F\_4BA0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																SIG_ANOTHER								SIG_LOGIC							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_PIGEON\_14\_2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved.
8–4 SIG_ANOTHER	select another signal for logic operation or as mask or counter tick event
3–0 SIG_LOGIC	logic operation with another signal sigout : final output signal of this generator mask : final mask of this generator this_sig : intermediate signal of this generator before logic operation

*Table continues on the next page...*



**EPDC\_PIGEON\_14\_2 field descriptions (continued)**

Field	Description
	other_masks : intermediate mask result of this generator before logic operation
	sig_another : signal selected other generators
0x0	<b>DIS</b> — no logic operation
0x1	<b>AND</b> — sigout = sig_another AND this_sig
0x2	<b>OR</b> — sigout = sig_another OR this_sig
0x3	<b>MASK</b> — mask = sig_another AND other_masks

## 22.4.98 Panel Interface Signal Generator Register 15\_0 (EPDC\_PIGEON\_15\_0)

parameters for timing signal generation

Address: 20F\_4000h base + BC0h offset = 20F\_4BC0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATE_MASK								MASK_CNT							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MASK_CNT				MASK_CNT_SEL				OFFSET				INC_SEL		POL	EN
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

**EPDC\_PIGEON\_15\_0 field descriptions**

Field	Description
31–24 STATE_MASK	state_mask = (FSIFBIFDIFE) and (LSILBILDILE) , select any combination of scan states as reference point for local counter to start ticking  0x1 <b>FS</b> — FRAME SYNC 0x2 <b>FB</b> — FRAME BEGIN 0x4 <b>FD</b> — FRAME DATA 0x8 <b>FE</b> — FRAME END 0x10 <b>LS</b> — LINE SYNC 0x20 <b>LB</b> — LINE BEGIN 0x40 <b>LD</b> — LINE DATA 0x80 <b>LE</b> — LINE END
23–12 MASK_CNT	When the global counter selected through MASK_CNT_SEL matches value in this reg, pigeon local counter start ticking. 0=disable
11–8 MASK_CNT_SEL	select global counters as mask condition, use together with MASK_CNT  0x0 <b>HSTATE_CNT</b> — pclk counter within one hscan state 0x1 <b>HSTATE_CYCLE</b> — pclk cycle within one hscan state 0x2 <b>VSTATE_CNT</b> — line counter within one vscan state 0x3 <b>VSTATE_CYCLE</b> — line cycle within one vscan state

*Table continues on the next page...*

**EPDC\_PIGEON\_15\_0 field descriptions (continued)**

Field	Description
	0x4 <b>FRAME_CNT</b> — frame counter 0x5 <b>FRAME_CYCLE</b> — frame cycle 0x6 <b>HCNT</b> — horizontal counter (pclk counter within one line ) 0x7 <b>VCNT</b> — vertical counter (line counter within one frame)
7–4 OFFSET	offset on pclk unit. 0=align with data, positive value means delay, minus value mean ahead. Supported range depends on panel mode
3–2 INC_SEL	event to increment local counter  0x0 <b>PCLK</b> — pclk 0x1 <b>LINE</b> — line start pulse 0x2 <b>FRAME</b> — frame start pulse 0x3 <b>SIG_ANOTHER</b> — use another signal as tick event
1 POL	polarity of signal output  0x0 <b>ACTIVE_HIGH</b> — normal signal (active high) 0x1 <b>ACTIVE_LOW</b> — inverted signal (active low)
0 EN	enable pigeon mode on this signal

## 22.4.99 Panel Interface Signal Generator Register 15\_1 (EPDC\_PIGEON\_15\_1)

parameters for timing signal generation

Address: 20F\_4000h base + BD0h offset = 20F\_4BD0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CLR_CNT																SET_CNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**EPDC\_PIGEON\_15\_1 field descriptions**

Field	Description
31–16 CLR_CNT	deassert signal output when counter match this value  0x0 <b>CLEAR_USING_MASK</b> — keep active until mask off
15–0 SET_CNT	assert signal output when counter match this value  0x0 <b>START_ACTIVE</b> — start as active

## 22.4.100 Panel Interface Signal Generator Register 15\_1 (EPDC\_PIGEON\_15\_2)

parameters for timing signal generation

Address: 20F\_4000h base + BE0h offset = 20F\_4BE0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																SIG_ANOTHER						SIG_LOGIC									
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPDC\_PIGEON\_15\_2 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved.
8–4 SIG_ANOTHER	select another signal for logic operation or as mask or counter tick event
3–0 SIG_LOGIC	<p>logic operation with another signal</p> <p>sigout : final output signal of this generator</p> <p>mask : final mask of this generator</p> <p>this_sig : intermediate signal of this generator before logic operation</p> <p>other_masks : intermediate mask result of this generator before logic operation</p> <p>sig_another : signal selected other generators</p> <p>0x0 <b>DIS</b> — no logic operation</p> <p>0x1 <b>AND</b> — sigout = sig_another AND this_sig</p> <p>0x2 <b>OR</b> — sigout = sig_another OR this_sig</p> <p>0x3 <b>MASK</b> — mask = sig_another AND other_masks</p>

## 22.4.101 EPDC Working Buffer Address for TCE (EPDC\_WB\_ADDR\_TCE)

EPDC Working Buffer Address used by TCE only

Address: 20F\_4000h base + C10h offset = 20F\_4C10h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	<div>ADDR</div>																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EPDC\_WB\_ADDR\_TCE field descriptions**

Field	Description
31–0 ADDR	Address for EPDC working buffer (only for TCE). This address must be aligned to a 64-bit double-word boundary.

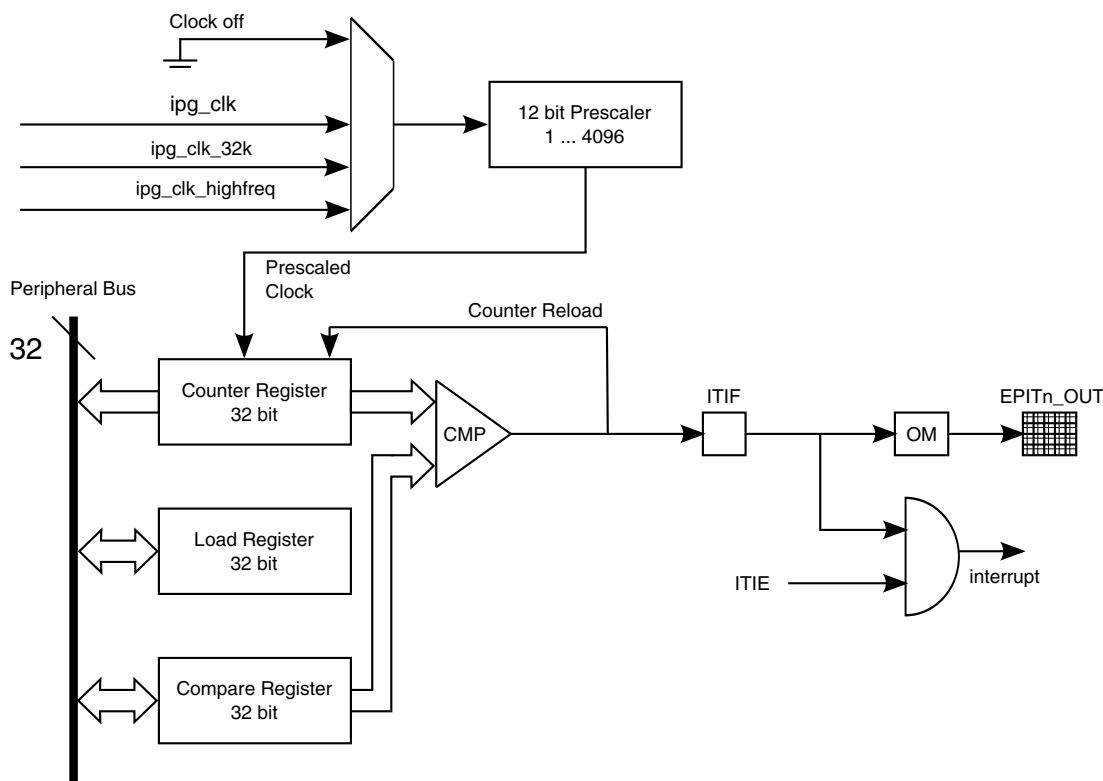
## Chapter 23

# Enhanced Periodic Interrupt Timer (EPIT)

### 23.1 Overview

EPIT is a 32-bit set-and-forget timer that is capable of providing precise interrupts at regular intervals with minimal processor intervention. EPIT begins counting after it is enabled by software.

The following figure shows the EPIT block diagram.



**Figure 23-1. EPIT block diagram**

### 23.1.1 EPIT features

EPIT has the following key features:

- 32-bit down counter with clock source selection
- 12-bit prescaler for division of input clock frequency
- Counter value that can be programmed on the fly
- Can be programmed to be active during low-power and debug modes
- Interrupt generation when counter reaches the compare value

### 23.1.2 EPIT modes and operations

EPIT supports the following modes: set-and-forget and free running. See the following sections for more information.

- [Operating in set-and-forget mode](#)
- [Operating in free-running mode](#)

See [Operations](#) for a description of the operations that EPIT supports.

## 23.2 External signals

The following table describes EPIT's I/O signals.

**Table 23-1. EPIT External Signals**

Signal	Description	Pad	Mode	Direction
EPIT1_OUT	Output 1 pin at chip boundary for indicating the occurrence of an output compare event through a specified transition.	PWM1	ALT6	O
		SD2_CMD	ALF4	
		SD3_CMD	ALT4	
EPIT2_OUT	Output 2 pin at chip boundary for indicating the occurrence of an output compare event through a specified transition.	LCD_DAT2	ALT2	O
		SD3_DAT3	ALT4	

## 23.3 Clocks

The table found here describes the clock sources for EPIT.

Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 23-2. EPIT Clocks**

Clock name	Clock Root	Description
ipg_clk	ipg_clk_root	Peripheral clock
ipg_clk_32k	ckil_sync_clk_root	Low-frequency reference clock (32kHz)
ipg_clk_highfreq	perclk_clk_root	High-frequency reference clock
ipg_clk_s	ipg_clk_root	Peripheral access clock

The clock that feeds the prescaler can be selected from among the following sources:

- **High-frequency reference clock (ipg\_clk\_highfreq)**

This clock is provided by the Clock Control Module (CCM). This clock remains on during low-power mode when the peripheral clock is turned off, allowing EPIT to use this clock in low-power mode. In normal mode, the CCM synchronizes this clock to ahb\_clk; in low-power mode, CCM switches to an unsynchronized version.

- **Low-frequency reference clock (ipg\_clk\_32k)**

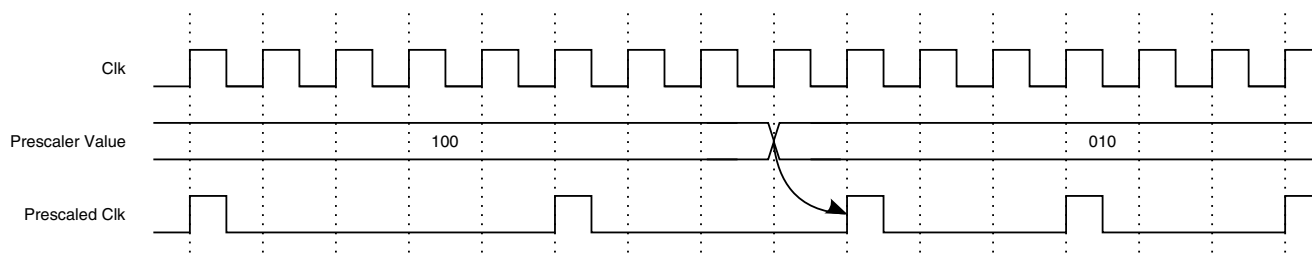
This 32 kHz reference clock is provided by the CCM. This clock remains on in low-power mode when the peripheral clock is turned off, so EPIT can use this clock during low-power mode. In normal mode, the CCM synchronizes this clock to ahb\_clk; in low-power mode, CCM switches to an unsynchronized version. This clock is derived from the external 32kHz crystal.

- **Peripheral clock (ipg\_clk)**

This is the peripheral clock (PER Clock) which is provided (and optionally gated) by the CCM. This clock is typically used in normal operations. In low-power modes, if the EPIT is programmed to be disabled (via STOPEN or WAITEN), then the peripheral clock can be switched off.

The clock input source is determined by the CLKSRC field in the control register. The clock input to the prescaler can also be disabled by setting CLKSRC to 0b00. **This field value should only be changed after first disabling the EPIT by clearing the EN bit in the EPIT\_EPITCR.** For other programming requirements that apply while changing clock source, refer section [Change of Clock Source](#).

The PRESCALER field in the control register is used to select the divide ratio of the input clock that drives the main counter. The prescaler can divide the input clock by a value between 1 and 4096. A change in the value of the PRESCALER field is immediately reflected on its output clock frequency. The following figure shows the timing for a change in the prescaler value.



**Figure 23-2. Prescaler Value Change Diagram**

## 23.4 Functional Description

This section provides a complete functional description of the block.

### 23.4.1 Operating modes

EPIT can operate in either set-and-forget or free-running mode. Use EPIT\_CR[RLD] to select the desired mode.

#### 23.4.1.1 Operating in set-and-forget mode

To select this mode of operation, set the RLD bit in the control register (EPIT\_CR).

In this mode, the counter obtains its data from the load register (EPIT\_LR); it cannot be written to directly from the block data bus. Whenever the counter reaches zero, the value in EPIT\_LR is loaded into the counter. This value is then decremented to zero.

To directly initialize the counter instead of waiting for the count to reach zero, set the EPIT counter overwrite enable bit (EPIT\_CR[IOVW]) and write to EPIT\_LR with the required initialization value.



### 23.4.1.2 Operating in free-running mode

To select this mode of operation, clear the RLD bit.

In this mode, the counter rolls over from 0000 0000h to FFFF FFFFh without reloading from the modulus register. After rolling over, the counter continues counting down.

To directly initialize the counter, set the EPIT counter overwrite enable bit (EPIT\_EPITCL[IOVW]) and write to EPIT\_EPITLR with the required initialization value.

### 23.4.2 Operations

EPIT has a single 32-bit down counter, which starts counting when the block is enabled by software.

The start value of the counter is loaded from the EPIT load register, which can be written to at any time by the processor. The value in the compare register determines the time that the interrupt occurs.

When EPIT is disabled (EN = 0), both the main counter and the prescaler counter freeze their count at their current count values. When EPIT is re-enabled (EN = 1), the ENMOD bit, which is a RW bit, decides the counter value:

- If ENMOD is set, the main counter is loaded with the load value (If RLD = 1)/ FFFF FFFFh (If RLD = 0) and the prescaler counter is reset (000h).
- If ENMOD is cleared, both main counter and prescaler counter restart counting from their frozen values.

If EPIT is programmed to be disabled in a low-power mode (STOP/WAIT), both the main counter and the prescaler counter freeze at their current count values when EPIT enters low-power mode. When EPIT exits the low-power mode, both the main counter and the prescaler counter start counting from their frozen values regardless of the ENMOD bit.

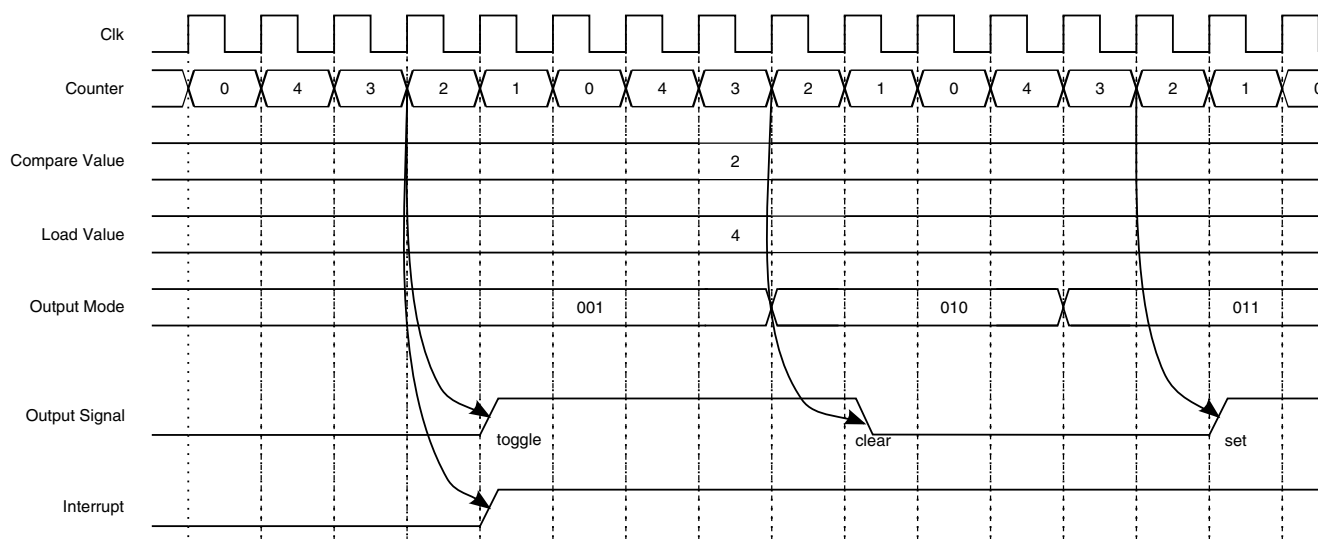
A hardware reset resets all EPIT registers to their respective reset values. There is a software reset which has the same effect on all registers except for the EN, ENMOD, STOPEN and WAITEN bits in the control register. The state of these bits are not affected by software reset. A software reset can be asserted even when the EPIT is disabled.

### 23.4.3 Compare Event

When the programmed value of EPITx\_EPITCMPR matches the value in EPITx\_EPITCNR, a compare status flag is set, and an interrupt is generated if the OCIE bit is set in the control register.

The compare output pin is set, cleared, toggled, or not affected at all depending on the setting of the output mode (OM) bits in the control register. If an interrupt is required at rollover (when the counter value reaches 0x0000\_0000 and the new value is loaded) then the compare register value should be set equal to the load register value in set-and-forget mode, or equal to 0xFFFF\_FFFF in free-running mode.

The following figure shows the timing for a compare event and interrupt.



**Figure 23-3. Compare Event and Interrupt Timing Diagram**

EPIT will generate a compare event in the next count if the EPITx\_CNR from the previous count equals the new EPITx\_CMPR configured before re-enabling the EPIT in the next count. Even in case a new start counter value was updated in EPITx\_LR before re-enabling the EPIT for the next round. To avoid this, configure the EPITx\_CMPR to previous EPITx\_CNR+1. Or, in set and forget mode, configure EPITx\_LR with IOVW=1, before disabling EPIT. Also can do an extra disable/enable iteration to clear OCIF and update EPITx\_CNR.

#### 23.4.3.1 Counter Value Overwrite

The EPIT counter value can be overwritten to acquire a desired value at any point of time. The procedure for this is to set the IOVW bit in the control register and then write the desired value into the load register.

This results in the load register acquiring that value and also the counter being overwritten with it. If the EPIT is running the counter resumes counting from the overwritten value.

### 23.4.3.2 Low-Power Mode Behavior

The EPIT timer's behavior in low-power modes depends on which clock source is being used.

If the selected clock source is available and the corresponding low-power enable bit is set, then the EPIT continues to function in the low-power mode. If the EPIT is programmed to be disabled in a low-power mode (STOP/WAIT), then main counter and the prescaler counter freeze at the current count values when the EPIT enters low-power mode. When the EPIT exits the low-power mode, both main counter and prescaler counter start counting from their frozen values irrespective of the ENMOD bit.

### 23.4.3.3 Debug Mode Behavior

In debug mode, the user has the option to run or halt the EPIT timers. If the DBGGEN bit is reset in the EPIT Control Register, the timer is halted.

When debug mode is exited, the timer operation reverts to what it was prior to entering debug mode.

## 23.5 Initialization/ Application Information

### 23.5.1 Change of Clock Source

The CLKSRC field in EPIT\_EPITCR determines the clock source. This field value should be changed only after disabling the EPIT (EN = 0).

Below is the software sequence which must be followed while changing clock source.

1. Disable the EPIT - set EN=0 in EPIT\_EPITCR.
2. Disable EPIT output - program OM=00 in the EPIT\_EPITCR.
3. Disable EPIT interrupts.
4. Program CLKSRC to desired clock source in EPIT\_EPITCR.
5. Clear the EPIT status register (EPIT\_EPITSR), that is, write "1" to clear (w1c).

6. Set ENMOD= 1 in the EPIT\_EPITCR, to bring the EPIT Counter to defined state (EPIT\_EPITLR value or 0xFFFF\_FFFF).
7. Enable EPIT - set (EN=1) in the EPIT\_EPITCR
8. Enable the EPIT interrupts.

## 23.6 EPIT Memory Map/Register Definition

The EPIT includes five user-accessible 32-bit registers. The following table summarizes these registers and their addresses.

Peripheral bus write access to the EPIT control register (EPITCR) and the EPIT load register (EPITLR) results in one cycle of wait state, while other valid peripheral bus accesses are with 0 wait state.

**EPIT memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20D_0000	Control register (EPIT1_CR)	32	R/W	0000_0000h	<a href="#">23.6.1/1040</a>
20D_0004	Status register (EPIT1_SR)	32	R/W	0000_0000h	<a href="#">23.6.2/1043</a>
20D_0008	Load register (EPIT1_LR)	32	R/W	FFFF_FFFFh	<a href="#">23.6.3/1044</a>
20D_000C	Compare register (EPIT1_CMPR)	32	R/W	0000_0000h	<a href="#">23.6.4/1044</a>
20D_0010	Counter register (EPIT1_CNR)	32	R	FFFF_FFFFh	<a href="#">23.6.5/1045</a>
20D_4000	Control register (EPIT2_CR)	32	R/W	0000_0000h	<a href="#">23.6.1/1040</a>
20D_4004	Status register (EPIT2_SR)	32	R/W	0000_0000h	<a href="#">23.6.2/1043</a>
20D_4008	Load register (EPIT2_LR)	32	R/W	FFFF_FFFFh	<a href="#">23.6.3/1044</a>
20D_400C	Compare register (EPIT2_CMPR)	32	R/W	0000_0000h	<a href="#">23.6.4/1044</a>
20D_4010	Counter register (EPIT2_CNR)	32	R	FFFF_FFFFh	<a href="#">23.6.5/1045</a>

### 23.6.2 Control register (EPITx\_CR)

The EPIT control register (EPIT\_CR) is used to configure the operating settings of the EPIT. It contains the clock division prescaler value and also the interrupt enable bit. Additionally, it contains other control bits which are described below.

Peripheral Bus Write access to EPIT Control Register (EPIT\_CR) results in one cycle of the wait state, while other valid peripheral bus accesses are with 0 wait state.

Address: Base address + 0h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0						CLKSRC		OM		STOPEN	0	WAITEN	DBGEN	IOVW	SWR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PRESCALAR												RLD	OCIE	ENMOD	EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## EPITx\_CR field descriptions

Field	Description
31–26 Reserved	This read-only field is reserved and always has the value 0.
25–24 CLKSRC	Select clock source These bits determine which clock input is to be selected for running the counter. This field value should only be changed when the EPIT is disabled by clearing the EN bit in this register. For other programming requirements while changing clock source, refer to <a href="#">Change of Clock Source</a> .  00 Clock is off 01 Peripheral clock 10 High-frequency reference clock 11 Low-frequency reference clock
23–22 OM	EPIT output mode. This bit field determines the mode of EPIT output on the output pin.  00 EPIT output is disconnected from pad 01 Toggle output pin 10 Clear output pin 11 Set output pin
21 STOPEN	EPIT stop mode enable. This read/write control bit enables the operation of the EPIT during stop mode. This bit is reset by a hardware reset and unaffected by software reset.  0 EPIT is disabled in stop mode 1 EPIT is enabled in stop mode
20 Reserved	This read-only field is reserved and always has the value 0.
19 WAITEN	This read/write control bit enables the operation of the EPIT during wait mode. This bit is reset by a hardware reset. A software reset does not affect this bit.  0 EPIT is disabled in wait mode 1 EPIT is enabled in wait mode
18 DBGEN	This bit is used to keep the EPIT functional in debug mode. When this bit is cleared, the input clock is gated off in debug mode. This bit is reset by hardware reset. A software reset does not affect this bit.

Table continues on the next page...

### EPITx\_CR field descriptions (continued)

Field	Description
	<p>0 Inactive in debug mode</p> <p>1 Active in debug mode</p>
17 IOVW	<p>EPIT counter overwrite enable. This bit controls the counter data when the modulus register is written. When this bit is set, all writes to the load register overwrites the counter contents and the counter starts subsequently counting down from the programmed value.</p> <p>0 Write to load register does not result in counter value being overwritten.</p> <p>1 Write to load register results in immediate overwriting of counter value.</p>
16 SWR	<p>Software reset. The EPIT is reset when this bit is set to 1. It is a self clearing bit. This bit is set when the block is in reset state and is cleared when the reset procedure is over. Setting this bit resets all the registers to their reset values, except for the EN, ENMOD, STOPEN, WAITEN and DBGEN bits in this control register</p> <p>0 EPIT is out of reset</p> <p>1 EPIT is undergoing reset</p>
15–4 PRESCALAR	<p>Counter clock prescaler value. This bit field determines the prescaler value by which the clock is divided before it goes to the counter</p> <p>0x000 Divide by 1</p> <p>0x001 Divide by 2...</p> <p>0xFFFF Divide by 4096</p>
3 RLD	<p>Counter reload control.</p> <p>This bit is cleared by hardware reset. It decides the counter functionality, whether to run in free-running mode or set-and-forget mode.</p> <p>0 When the counter reaches zero it rolls over to 0xFFFF_FFFF (free-running mode)</p> <p>1 When the counter reaches zero it reloads from the modulus register (set-and-forget mode)</p>
2 OCIE	<p>Output compare interrupt enable.</p> <p>This bit enables the generation of interrupt on occurrence of compare event.</p> <p>0 Compare interrupt disabled</p> <p>1 Compare interrupt enabled</p>
1 ENMOD	<p>EPIT enable mode.</p> <p>When EPIT is disabled (EN=0), both main counter and prescaler counter freeze their count at current count values. ENMOD bit is a r/w bit that determines the counter value when the EPIT is enabled again by setting EN bit. If ENMOD bit is set, then main counter is loaded with the load value (If RLD=1)/ 0xFFFF_FFFF (If RLD=0) and prescaler counter is reset, when EPIT is enabled (EN=1). If ENMOD is programmed to 0 then both main counter and prescaler counter restart counting from their frozen values when EPIT is enabled (EN=1). If EPIT is programmed to be disabled in a low-power mode (STOP/WAIT/DEBUG), then both the main counter and the prescaler counter freeze at their current count values when EPIT enters low-power mode. When EPIT exits the low-power mode, both main counter and prescaler counter start counting from their frozen values irrespective of the ENMOD bit. This bit is reset by a hardware reset. A software reset does not affect this bit.</p> <p>0 Counter starts counting from the value it had when it was disabled.</p> <p>1 Counter starts count from load value (RLD=1) or 0xFFFF_FFFF (If RLD=0)</p>
0 EN	<p>This bit enables the EPIT. EPIT counter and prescaler value when EPIT is enabled (EN = 1), is dependent upon ENMOD and RLD bit as described for ENMOD bit. It is recommended that all registers be properly programmed before setting this bit. This bit is reset by a hardware reset. A software reset does not affect this bit.</p>

Table continues on the next page...

**EPITx\_CR field descriptions (continued)**

Field	Description
0	EPIT is disabled
1	EPIT is enabled

**23.6.3 Status register (EPITx\_SR)**

The EPIT status register (EPIT\_SR) has a single status bit for the output compare event. The bit is a write 1 to clear bit.

Address: Base address + 4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															OCIF
W																w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**EPITx\_SR field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 OCIF	Output compare interrupt flag. This bit is the interrupt flag that is set when the content of counter equals the content of the compare register (EPIT_CMPR). The bit is a write 1 to clear bit.  0 Compare event has not occurred 1 Compare event occurred

## 23.6.4 Load register (EPITx\_LR)

The EPIT load register (EPIT\_LR) contains the value that is to be loaded into the counter when EPIT counter reaches zero if the RLD bit in EPIT\_CR is set. If the IOVW bit in the EPIT\_CR is set then a write to this register overwrites the value of the EPIT counter register in addition to updating this registers value. This overwrite feature is active even if the RLD bit is not set.

Address: Base address + 8h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W	LOAD																															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

### EPITx\_LR field descriptions

Field	Description
31–0 LOAD	Load value. Value that is loaded into the counter at the start of each count cycle.

## 23.6.5 Compare register (EPITx\_CMPR)

The EPIT compare register (EPIT\_CMPR) holds the value that determines when a compare event is generated.

Address: Base address + Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W	COMPARE																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### EPITx\_CMPR field descriptions

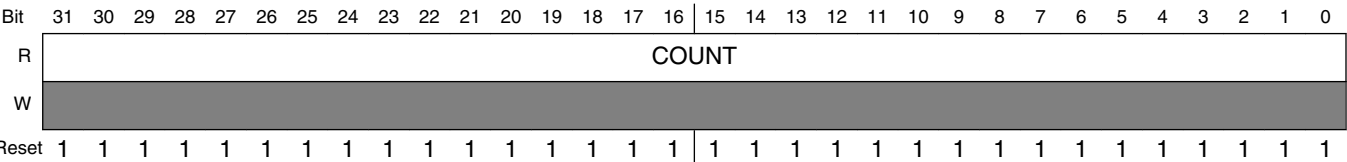
Field	Description
31–0 COMPARE	Compare Value. When the counter value equals this bit field value a compare event is generated.



23.6.6 Counter register (EPITx\_CNR)

The EPIT counter register (EPIT\_CNR) contains the current count value and can be read at any time without disturbing the counter. This is a read-only register and any attempt to write into it generates a transfer error. But if the IOVW bit in EPIT\_CR is set, the value of this register can be overwritten with a write to EPIT\_LR. This change is reflected when this register is subsequently read.

Address: Base address + 10h offset



EPITx\_CNR field descriptions

Field	Description
31–0 COUNT	Counter value. This contains the current value of the counter.



# Chapter 24

## Fast Ethernet Controller (FEC)

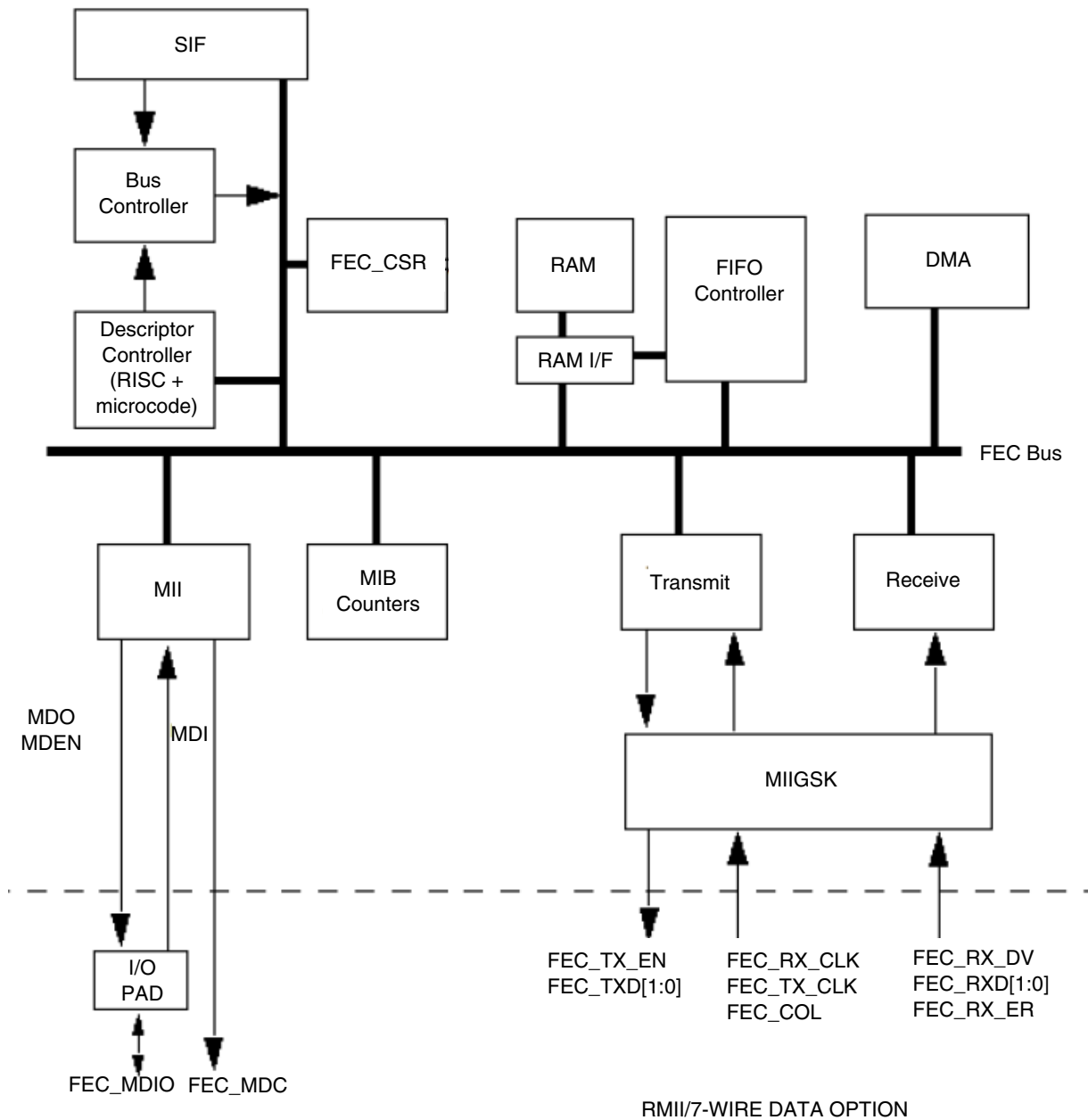
### 24.1 Overview

This chapter provides a feature-set overview, a functional block diagram, and transceiver connection information for the 10- and 100-Mbps reduced media independent interfaces (RMII) and the 7-wire serial interface.

Detailed functional descriptions and application/initialization information are included.

The Fast Ethernet controller (FEC) is designed to support both 10- and 100-Mbps Ethernet/IEEE 802.3 networks. An external transceiver interface and transceiver function are required to complete the interface to the media. The FEC supports different standard physical interfaces (MAC-PHY) for connection to an external Ethernet transceiver.

The FEC is implemented with a combination of hardware and microcode. The following figure is a block diagram of the FEC.



**Figure 24-1. FEC Block Diagram**

The following section provides information regarding the FEC submodules.

- The descriptor controller is a RISC-based controller that provides the following functions in the FEC:
  - Initialization (those internal registers not initialized by the user or hardware)
  - High-level control of the DMA channels (initiating DMA transfers)
  - Interpreting buffer descriptors
  - Address recognition for receive frames
  - Random number generation for transmit collision backoff timer

**NOTE**

This DMA engine is for the transfer of FEC data only and is not related to the system DMA controller.

- The RAM is the focal point of all data flow in the FEC and is divided into transmit and receive FIFOs. The FIFO boundaries are programmable using the FEC\_FRSR register. User data flows to and from the DMA block, from and to the receive/transmit FIFOs. Transmit data flows from the transmit FIFO to the transmit block, and receive data flows from the receive block to the receive FIFO.
- The user controls the FEC by writing, through the slave interface (SIF) submodule, to control registers located in each block. The control and status register (FEC\_CSR) provides global control (for example, Ethernet reset and enable) and interrupt handling registers.
- The MII provides a serial channel for control/status communication with the external physical layer device (transceiver). This serial channel consists of the management data clock and management data input/output lines of the MII interface (FEC\_MDC and FEC\_MDIO, respectively).
- The DMA provides multiple channels allowing transmit data, transmit descriptor, receive data and receive descriptor accesses to run independently.
- The transmit and receive blocks provide the Ethernet MAC functionality (with some assistance from microcode).
- The message information block (MIB) maintains counters for a variety of network events and statistics. It is not necessary for operation of the FEC but provides valuable counters for network management. The counters supported are the RMON (RFC 1757) Ethernet statistics group and some of the IEEE 802.3 counters. See Message Information Block (MIB) Counters Memory Map, for more information.
- The MII gasket block (MIIGSK) provides an interface between the MII (Media Independent Interface specified by IEEE 802.3 standard) I/F and RMII (low pin count Reduced Media Independent Interface as specified by the RMII Consortium™ standard) I/F.

## 24.1.1 Features

The FEC incorporates the following features:

- Support for different Ethernet physical interfaces:
  - 10-Mbps and 100-Mbps RMII
  - 10-Mbps 7-wire interface (industry standard)
  - Does not support 10/100-Mbps MII
- IEEE 802.3 full-duplex flow control
- Programmable max frame length supports IEEE 802.1 VLAN tags and priority

- Support for full-duplex operation (200Mbps throughput) with a minimum system clock rate of 50 MHz
- Support for half-duplex operation (100Mbps throughput) with a minimum system clock rate of 25 MHz
- Retransmission from transmit FIFO following a collision (no processor bus utilization)
- Automatic internal flushing of the receive FIFO for runts (collision fragments) and address recognition rejects (no processor bus utilization)
- Address recognition
  - Frames with broadcast address can be always accepted or always rejected
  - Exact match for single 48-bit individual (unicast) address
  - Hash (64-bit hash) check of individual (unicast) addresses
  - Hash (64-bit hash) check of group (multicast) addresses
  - Promiscuous mode

## 24.2 External Signals

Please see [Network Interface Options](#) for external signal information.

## 24.3 Clocks

The following table describes the clock sources for FEC. Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 24-1. FEC Clocks**

Clock name	Clock Root	Description
fec_clk	ipg_clk_root	Peripheral clock
fec_hclk	ahb_clk_root	Module / Bus clock

## 24.4 Modes of Operation

The primary operational modes are described in this section.

## 24.4.1 Full- and Half-Duplex Operation

Full-duplex mode is intended for use on point to point links between switches or end node to switch. Half-duplex mode is used in connections between an end node and a repeater or between repeaters.

Selection of the duplex mode is controlled by the FEC\_TCR[FDEN] bit. When configured for full-duplex mode, flow control can be enabled, which is effected by the FEC\_TCR[RFC\_PAUSE], FEC\_TCR[TFC\_PAUSE], and FEC\_RCR[FCE] bits. See [Full-Duplex Flow Control](#), for more details.

## 24.4.2 Interface Options

The following interface options are supported. A detailed discussion of the interface configurations is provided in [Network Interface Options](#).

### 24.4.2.1 10 Mbps and 100 Mbps RMII Interface

RMII is a reduced MII interface specified by the RMII Consortium™ standard. The purpose of this interface is to provide a low cost alternative to the MII, with 8 pin interface instead of 16 (Not including MDC, MDIO pins).

The FEC uses a gasket (MIIGSK) to support RMII. It will operate in RMII mode by setting FEC\_MIIGSK\_CFGR[I/F\_MODE] = 01, while setting FEC\_MIIGSK\_CFGR[I/F\_MODE] = 00 will disable RMII mode. The speed selection in RMII Mode is supported by configuring the FEC\_MIIGSK\_CFGR[FRCONT]. The serial management interface is still available in RMII mode.

### 24.4.2.2 10-Mbps 7-Wire Interface Operation

The FEC supports a 7-wire interface as used by many 10 Mbps ethernet transceivers. The FEC\_RCR[MII\_MODE] bit controls this functionality. If this bit is cleared, the MII mode is disabled and the 10 Mbps, 7-wire mode is enabled.

## 24.4.3 Address Recognition Options

The address options supported are promiscuous, broadcast reject, individual address (hash or exact match), and multicast hash match. Address recognition options are discussed in detail in [Ethernet Address Recognition](#).

## 24.4.4 Internal Loopback

Two levels of internal loopback modes are supported. The first level, which will loop data back before they enter MIIGSK, is selected by FEC\_RCR[LOOP]. The second level, which will loop data back inside MIIGSK RMII domain, is selected by setting FEC\_MIIGSK\_CFGR[LBMODE] and FEC\_MIIGSK\_CFGR[I/F\_MODE] = 01. Loopback mode is discussed in detail in [Internal and External Loopback](#).

## 24.5 Functional Description

This section provides a detailed description of the functions of the FEC.

- Network interface options
- Frame transmission and reception
- Full-duplex flow control
- Internal and external loopback

### 24.5.1 Network Interface Options

The FEC supports an RMII interface for 10/100 Mbps Ethernet and a 7-wire serial interface for 10 Mbps Ethernet.

The interface mode is selected by FEC\_MIIGSK\_CFGR[I/F\_MODE] together with the FEC\_RCR[MII\_MODE] bit as shown in the table below.

**Table 24-2. Interface Mode Selection**

FEC_MIIGSK_CFGR[I/F_MODE]	FEC_RCR[MII_MODE]	Interface Mode Selected
00	0	7-Wire
01	1	RMII

In RMII mode, a reduced set of 10 signals are used. These signals are shown in [Table 24-3](#).

**Table 24-3. RMII Mode Signal Configuration**

Signal Description	FEC Signal	Pad	Mode	Direction
Synchronous clock reference (REF_CLK)	FEC_TX_CLK	FEC_TX_CLK	Alt0	In
Transmit Enable	FEC_TX_EN	FEC_TX_EN	Alt0	Out

*Table continues on the next page...*



**Table 24-3. RMII Mode Signal Configuration (continued)**

Signal Description	FEC Signal	Pad	Mode	Direction
Transmit Data	FEC_TXD[1:0]	FEC_TXD0/1	Alt0	Out
Carrier Sense/Receive Data Valid (CRS_DV)	FEC_RX_DV	FEC_CRD_DV	Alt0	In
Receive Data	FEC_RXD[1:0]	FEC_RXD0/1	Alt0	In
Receive Error	FEC_RX_ER	FEC_RX_ER	Alt0	In
Management Data Clock	FEC_MDC	FEC_MDC	Alt0	Out
Management Data Input/Output	FEC_MDIO	FEC_MDIO	Alt0	I/O

The 7-wire serial interface operates in what is generally referred to as AMD mode. 7-wire mode connections to the external transceiver are shown in [Table 24-4](#).

**Table 24-4. 7-Wire Mode Signal Configuration**

Signal Description	FEC Signal	Pad	Mode	Direction
Transmit clock	FEC_TX_CLK	FEC_TX_CLK	Alt0	In
Transmit enable	FEC_TX_EN	FEC_TX_EN	Alt0	Out
Transmit data	FEC_TXD[0]	FEC_TXD0	Alt0	Out
Collision	FEC_COL	FEC_RXD1	Alt6	In
Receive clock	FEC_RX_CLK	FEC_TXD1	Alt6	In
Receive data valid	FEC_RX_DV	FEC_CRD_DV	Alt0	In
Receive data	FEC_RXD[0]	FEC_RXD0	Alt0	In

## 24.5.2 FEC Frame Transmission

The Ethernet transmitter is designed to work with almost no intervention from software.

After FEC\_ECR[ETHER\_EN] is set to 1 and data appears in the transmit FIFO, the FEC is able to transmit on to the network.

When the transmit FIFO fills to the watermark (defined by the FEC\_TFWR register), the FEC transmit logic asserts FEC\_TX\_EN and starts transmitting the preamble (PA) sequence, the start frame delimiter (SFD), and then the frame information from the FIFO. However, the controller postpones transmission if the network is busy (that is, the carrier sense signal FEC\_CRD is asserted). Before transmitting, the controller waits for carrier sense to become inactive, then determines if carrier sense remains inactive for 60 bit periods. If so, the transmission begins after waiting an additional 36 bit periods (96 bit periods after carrier sense originally became inactive). See [Transmission Error Handling](#) for more details.

If a collision occurs during transmission of a frame in half-duplex mode, the Ethernet controller follows the specified backoff procedures (see [Collision Handling](#)) and attempts to retransmit the frame until the retry limit is reached. The transmit FIFO stores at least the first 64 bytes of the transmit frame, so that they do not have to be retrieved from system memory in case of a collision. This improves bus utilization and latency in case immediate retransmission is necessary.

When all the frame data has been transmitted, if the TC bit is set in the transmit frame control word then a 32-bit cyclic redundancy check (CRC) known as the frame check sequence (FCS) is appended. If the ABC bit is set in the transmit frame control word, a bad CRC is appended to the frame data regardless of the TC bit value. Following the transmission of the CRC, the Ethernet controller writes the frame status information to the MIB block. Short frames are automatically padded by the transmit logic (if the TC bit in the transmit buffer descriptor for the end of frame buffer is set to 1).

Both buffer (TXB) and frame (TXF) interrupts can be generated as determined by the settings in the FEC\_EIMR.

The transmit error interrupts are HBERR, BABT, LATE\_COL, COL\_RETRY\_LIM, and XFIFO\_UN. If the transmit frame length exceeds MAX\_FL bytes the BABT interrupt is asserted: however, the entire frame is transmitted (no truncation).

Transmission is paused by setting the graceful transmit stop (GTS) bit in the FEC\_TCR register. When the FEC\_TCR[GTS] is set to 1, the FEC transmitter stops immediately if transmission is not in progress; otherwise, it continues transmission until the current frame either finishes or terminates with a collision. After the transmitter has stopped the GRA (graceful stop complete) interrupt is asserted. If FEC\_TCR[GTS] is cleared, the FEC resumes transmission with the next frame.

The Ethernet controller transmits bytes LSB first.

### **24.5.2.1 Transmit Inter-Packet Gap (IPG) Time**

The minimum inter-packet gap (IPG) time for back-to-back transmission is 96 bit periods. After completing a transmission or after the backoff algorithm completes, the transmitter waits for carrier sense to be negated before starting its 96-bit-period IPG counter. Frame transmission begins 96 bit periods after carrier sense is negated if it stays negated for at least 60 bit periods. If carrier sense is asserted during the last 36 bit periods, it is ignored and a collision occurs.

### 24.5.2.2 Collision Handling

If a collision occurs during frame transmission, the Ethernet controller continues the transmission for at least 32 bit periods, transmitting a jam pattern consisting of 32 ones.

If the collision occurs during the preamble sequence, the jam pattern is sent after the end of the preamble sequence.

If a collision occurs within 512 bit periods, the retry process is initiated. The transmitter waits a random number of slot periods, where one slot period is 512 bit periods. If a collision occurs after 512 bit periods, then no retransmission is performed and the end of frame buffer is closed with a late collision (LC) error indication.

### 24.5.2.3 Transmission Error Handling

The Ethernet controller reports frame transmission error conditions using the FEC RxBDs, the FEC\_EIR register, and the MIB block counters.

There are four types of transmission errors:

- Transmitter underrun
- Retransmission attempts limit expired
- Late collision
- Heartbeat

The FEC's procedures for handling these errors are described in the following subsections.

#### 24.5.2.3.1 Transmitter Underrun

If this error occurs, the FEC sends 32 bits that ensure a CRC error, then stops transmitting. All remaining buffers for that frame are then flushed and closed. The UN bit is set in the FEC\_EIR and the UN interrupt is asserted (if enabled in the FEC\_EIMR register). The FEC then continues to the next transmit buffer descriptor and begin transmitting the next frame.

#### 24.5.2.3.2 Retransmission Attempts Limit Expired

When this error occurs, the FEC terminates transmission. All remaining buffers for that frame are flushed and closed, and the RL bit is set in the FEC\_EIR. The FEC then continues to the next transmit buffer descriptor and begin transmitting the next frame.

The "RL" interrupt is asserted if enabled in the FEC\_EIMR register.

### 24.5.2.3.3 Late Collision

When a collision occurs after the slot time (512 bits starting at the preamble), the FEC terminates transmission. All remaining buffers for that frame are flushed and closed, and the LC bit is set in the FEC\_EIR register. The FEC then continues to the next transmit buffer descriptor and begin transmitting the next frame.

The LC interrupt is asserted if enabled in the FEC\_EIMR register.

### 24.5.2.3.4 Heartbeat

Some transceivers have a self-test feature called "heartbeat" or "signal quality error." To signify a good self-test, the transceiver indicates a collision to the FEC within 4 microseconds after completion of a frame transmitted by the Ethernet controller. This indication of a collision does not imply a real collision error on the network, but is rather an indication that the transceiver still seems to be functioning properly. This is called the heartbeat condition.

If the HBC bit is set in the FEC\_TCR register and the heartbeat condition is not detected by the FEC after a frame transmission, then a heartbeat error occurs. When this error occurs, the FEC closes the buffer, sets the HB bit in the FEC\_EIR register, and generates the HBERR interrupt if it is enabled.

## 24.5.3 FEC Frame Reception

The FEC receiver is designed to work with almost no intervention from the host and can perform address recognition, CRC checking, short frame checking, and maximum frame length checking.

When the driver enables the FEC receiver by setting FEC\_ECR[ETHER\_EN] to 1, it immediately starts processing receive frames. When FEC\_RX\_DV asserts, the receiver first checks for a valid PA/SFD header. If the PA/SFD is valid, it is stripped and the frame is processed by the receiver. If a valid PA/SFD is not found, the frame is ignored.

In serial mode, the first 16 bit periods of RX\_D0 following assertion of FEC\_RX\_DV are ignored. Following the first 16 bit periods the data sequence is checked for alternating 1/0s. If a 0b11 or 0b00 data sequence is detected during bit periods 17 to 21, the remainder of the frame is ignored. After bit period 21, the data sequence is monitored for a valid SFD (11). If a 0b00 is detected, the frame is rejected. When a 0b11 is detected, the PA/SFD sequence is complete.

In RMII mode, the receiver checks for at least one byte matching the SFD. Zero or more PA bytes can occur, but if a 0b00 bit sequence is detected prior to the SFD byte, the frame is ignored.

After the first 6 bytes of the frame have been received, the FEC performs address recognition on the frame.

After a collision window (64 bytes) of data has been received, and if address recognition has not rejected the frame, the receive FIFO is signaled that the frame is accepted and can be passed on to the DMA. If the frame is a runt (due to collision) or is rejected by address recognition, the receive FIFO is notified to reject the frame. Thus no collision fragments are presented to the user except late collisions, which indicate serious LAN problems.

During reception, the Ethernet controller checks for various error conditions and after the entire frame is written into the FIFO, a 32-bit frame status word is written into the FIFO. This status word contains the M, BC, MC, LG, NO, CR, OV and TR status bits, and the frame length. See [Reception Error Handling](#) for more details.

Receive Buffer (RXB) and Frame Interrupts (RXF) can be generated if enabled by the FEC\_EIMR register. The only receive error interrupt is babbling receiver error (BABR). Receive frames are not truncated if they exceed the max frame length (MAX\_FL); however, the BABR interrupt occurs and the LG bit in the receive buffer descriptor (RxBD) is set. See [Ethernet Receive Buffer Descriptor \(RxBD\)](#) for more details.

When the receive frame is complete, the FEC sets the L bit in the RxBD, writes the other frame status bits into the RxBD, and clears the E bit. The Ethernet controller next generates a maskable interrupt (RXF bit in FEC\_EIR, maskable by RXF bit in FEC\_EIMR), indicating that a frame has been received and is in memory. The Ethernet controller then waits for a new frame.

The Ethernet controller receives serial data LSB first.

### 24.5.3.1 Receive Inter-Packet Gap (IPG) Time

The receiver receives back-to-back frames with a minimum spacing of 28-bit periods. If an inter-packet gap between receive frames is less than 28 bit periods, the second frame may be discarded by the receiver.

### 24.5.3.2 Ethernet Address Recognition

Address recognition is accomplished through the use of the receive block and microcode running on the microcontroller.

The flowchart shown in [Figure 24-2](#) illustrates the address recognition decisions made by the receive block, while [Figure 24-3](#) illustrates the decisions made by the microcontroller.

The FEC filters the received frames based on destination address (DA) type - individual (unicast), group (multicast), or broadcast (all-ones group address). The difference between an individual address and a group address is determined by the I/G bit in the DA field.

If the DA is a broadcast address, and broadcast reject (FEC\_RCR[BC\_REJ]) is cleared, then the frame is accepted unconditionally, as shown in [Figure 24-2](#). Otherwise, if the DA is not a broadcast address, then the microcontroller runs the address recognition subroutine, as shown in [Figure 24-3](#).

If the DA is a group (multicast) address and flow control is disabled, then the microcontroller performs a group hash table lookup using the 64-entry hash table programmed in FEC\_GAUR and FEC\_GALR. If a hash match occurs, the receiver accepts the frame. The hash algorithm is described in [Hash Algorithm](#).

If flow control is enabled, the microcontroller does an exact address match check between the DA and the designated PAUSE DA (01:80:C2:00:00:01). If the receive block determines that the received frame is a valid pause frame, then the frame is rejected.

### **NOTE**

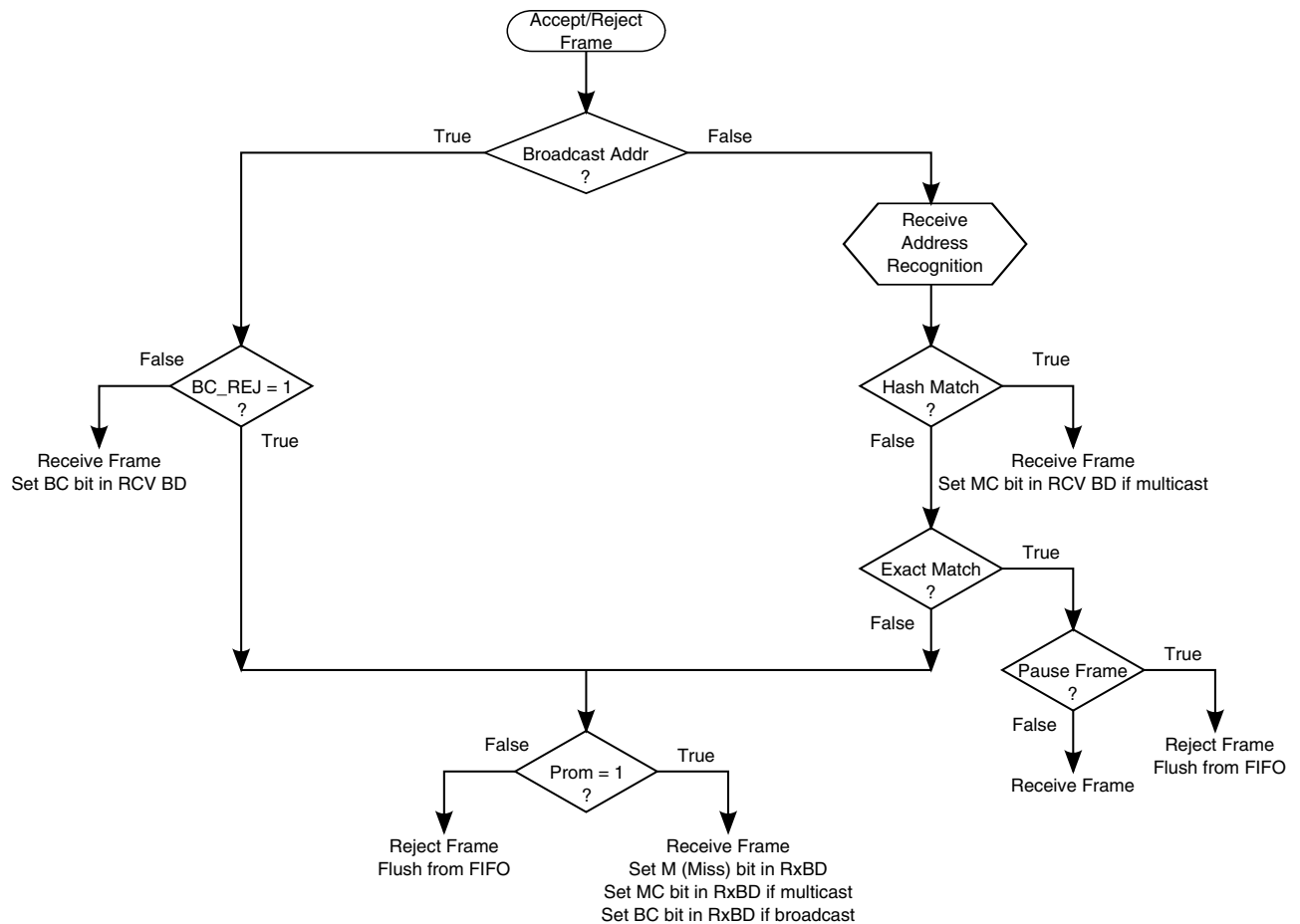
The receiver detects a pause frame with the DA field set to either the designated PAUSE DA or the unicast physical address.

If the DA is the individual (unicast) address, the microcontroller performs an individual exact match comparison between the DA and 48-bit physical address that the user programs in the FEC\_PALR and FEC\_PAUR registers. If an exact match occurs, the frame is accepted; otherwise, the microcontroller does an individual hash table lookup using the 64-entry hash table programmed in registers, FEC\_IAUR and FEC\_IALR (the hash algorithm is described in [Hash Algorithm](#)). In the case of an individual hash match, the frame is accepted. Again, the receiver accepts or rejects the frame based on pause frame detection, shown in [Figure 24-2](#).

If neither a hash match (group or individual), nor an exact match occur, then if promiscuous mode is enabled (FEC\_RCR[PROM] = 1), the frame is accepted and the MISS bit in the receive buffer descriptor is set. Otherwise, the frame is rejected.

Similarly, if the DA is a broadcast address, broadcast reject (FEC\_RCR[BC\_REJ]) is asserted, and promiscuous mode is enabled, then the frame is accepted and the MISS bit in the receive buffer descriptor is set. Otherwise, the frame is rejected.

In general, when a frame is rejected, it is flushed from the FIFO.



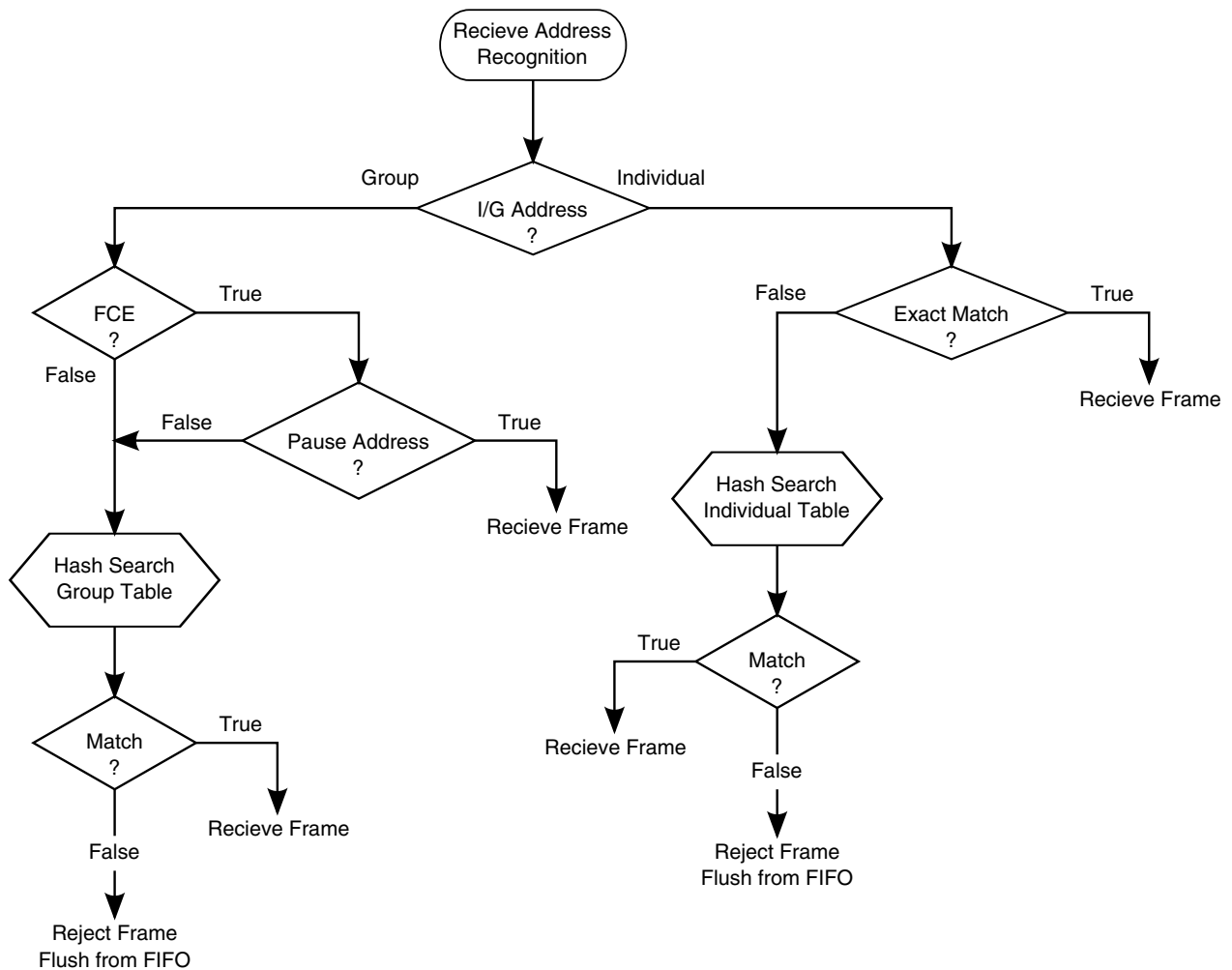
## NOTES:

BC\_REJ - Field in RCR register (BroadCast REject)

PROM-Field in RCR register (PROMiscuous mode)

Pause Frame - Valid PAUSE frame received

**Figure 24-2. Ethernet Address Recognition-Receive Block Decisions**



## NOTES:

FCE - field in RCR register (Flow Control Enable)

I/G - Individual/Group bit in Destination Address (least significant bit in first byte received in MAC frame)

**Figure 24-3. Ethernet Address Recognition-Microcode Decisions****24.5.3.2.1 Hash Algorithm**

The hash table algorithm used in the group and individual hash filtering operates as follows. The 48-bit destination address is mapped into one of 64 bits, which are represented by 64 bits stored in FEC\_GAUR, FEC\_GALR (group address hash match) or FEC\_IAUR, FEC\_IALR (individual address hash match). This mapping is performed by passing the 48-bit address through the on-chip 32-bit CRC generator and selecting the six most significant bits of the CRC-encoded result to generate a number between 0 and 63. The MSB of the CRC result selects FEC\_GAUR (MSB = 1) or FEC\_GALR (MSB = 0). The least significant five bits of the hash result select the bit within the selected register. If the CRC generator selects a bit that is set in the hash table, the frame is accepted; otherwise, it is rejected. For example, if eight group addresses are stored in the hash table



and random group addresses are received, the hash table prevents roughly 56/64 (87.5%) of the group address frames from reaching memory. Those that do reach memory must be further filtered by the processor to determine if they truly contain one of the eight desired addresses.

The effectiveness of the hash table declines as the number of addresses increases.

The hash table registers must be initialized by the user. The CRC32 polynomial to use in computing the hash is shown in the following equation:

$$X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} + X^{10} + X^8 + X^7 + X^5 + X^4 + X^2 + X + 1$$

Table 24-5 shows example destination addresses and corresponding hash values is included below for reference.

**Table 24-5. Destination Address to 6-Bit Hash**

48-bit DA (in hex)	6-bit Hash (in hex)	Hash Decimal Value
65:FF:FF:FF:FF:FF	0x0	0
55:FF:FF:FF:FF:FF	0x1	1
15:FF:FF:FF:FF:FF	0x2	2
35:FF:FF:FF:FF:FF	0x3	3
B5:FF:FF:FF:FF:FF	0x4	4
95:FF:FF:FF:FF:FF	0x5	5
D5:FF:FF:FF:FF:FF	0x6	6
F5:FF:FF:FF:FF:FF	0x7	7
DB:FF:FF:FF:FF:FF	0x8	8
FB:FF:FF:FF:FF:FF	0x9	9
BB:FF:FF:FF:FF:FF	0xA	10
8B:FF:FF:FF:FF:FF	0xB	11
0B:FF:FF:FF:FF:FF	0xC	12
3B:FF:FF:FF:FF:FF	0xD	13
7B:FF:FF:FF:FF:FF	0xE	14
5B:FF:FF:FF:FF:FF	0xF	15
27:FF:FF:FF:FF:FF	0x10	16
07:FF:FF:FF:FF:FF	0x11	17
57:FF:FF:FF:FF:FF	0x12	18
77:FF:FF:FF:FF:FF	0x13	19

*Table continues on the next page...*

**Table 24-5. Destination Address to 6-Bit Hash (continued)**

48-bit DA (in hex)	6-bit Hash (in hex)	Hash Decimal Value
F7:FF:FF:FF:FF:FF	0x14	20
C7:FF:FF:FF:FF:FF	0x15	21
97:FF:FF:FF:FF:FF	0x16	22
A7:FF:FF:FF:FF:FF	0x17	23
99:FF:FF:FF:FF:FF	0x18	24
B9:FF:FF:FF:FF:FF	0x19	25
F9:FF:FF:FF:FF:FF	0x1A	26
C9:FF:FF:FF:FF:FF	0x1B	27
59:FF:FF:FF:FF:FF	0x1C	28
79:FF:FF:FF:FF:FF	0x1D	29
29:FF:FF:FF:FF:FF	0x1E	30
19:FF:FF:FF:FF:FF	0x1F	31
D1:FF:FF:FF:FF:FF	0x20	32
F1:FF:FF:FF:FF:FF	0x21	33
B1:FF:FF:FF:FF:FF	0x22	34
91:FF:FF:FF:FF:FF	0x23	35
11:FF:FF:FF:FF:FF	0x24	36
31:FF:FF:FF:FF:FF	0x25	37
71:FF:FF:FF:FF:FF	0x26	38
51:FF:FF:FF:FF:FF	0x27	39
7F:FF:FF:FF:FF:FF	0x28	40
4F:FF:FF:FF:FF:FF	0x29	41
1F:FF:FF:FF:FF:FF	0x2A	42
3F:FF:FF:FF:FF:FF	0x2B	43
BF:FF:FF:FF:FF:FF	0x2C	44
9F:FF:FF:FF:FF:FF	0x2D	45
DF:FF:FF:FF:FF:FF	0x2E	46
EF:FF:FF:FF:FF:FF	0x2F	47
93:FF:FF:FF:FF:FF	0x30	48
B3:FF:FF:FF:FF:FF	0x31	49
F3:FF:FF:FF:FF:FF	0x32	50
D3:FF:FF:FF:FF:FF	0x33	51
53:FF:FF:FF:FF:FF	0x34	52
73:FF:FF:FF:FF:FF	0x35	53
23:FF:FF:FF:FF:FF	0x36	54
13:FF:FF:FF:FF:FF	0x37	55
3D:FF:FF:FF:FF:FF	0x38	56
0D:FF:FF:FF:FF:FF	0x39	57
5D:FF:FF:FF:FF:FF	0x3A	58

*Table continues on the next page...*

**Table 24-5. Destination Address to 6-Bit Hash (continued)**

48-bit DA (in hex)	6-bit Hash (in hex)	Hash Decimal Value
7D:FF:FF:FF:FF:FF	0x3B	59
FD:FF:FF:FF:FF:FF	0x3C	60
DD:FF:FF:FF:FF:FF	0x3D	61
9D:FF:FF:FF:FF:FF	0x3E	62
BD:FF:FF:FF:FF:FF	0x3F	63

### 24.5.3.3 Reception Error Handling

The Ethernet controller reports frame reception error conditions using the FEC RxBDs, the FEC\_EIR register, and the MIB counters.

There are five types of reception errors:

- Overrun
- Non-octet (dribbling bits)
- CRC
- Frame-length violation
- Truncation

These are described in the following subsections.

#### 24.5.3.3.1 Overrun

If the receive block has data to put into the receive FIFO and the receive FIFO is full, the FEC sets the OV bit in the RxBD. All subsequent data in the frame is discarded, and subsequent frames can also be discarded until the receive FIFO is serviced by the DMA and space is made available. At this point the receive frame/status word is written into the FIFO with the OV bit set. This frame must be discarded by the driver.

#### 24.5.3.3.2 Non-Octet (Dribbling Bits)

The Ethernet controller handles up to seven dribbling bits when the receive frame terminates past a non-octet aligned boundary. Dribbling bits are not used in the CRC calculation. If there is a CRC error, then the frame non-octet aligned (NO) error is reported in the RxBD. If there is no CRC error, then no error is reported.

### 24.5.3.3.3 CRC

When a CRC error occurs with no dribble bits, the FEC closes the buffer and sets the CR bit in the RxBD. CRC checking cannot be disabled, but the CRC error can be ignored if checking is not required.

### 24.5.3.3.4 Frame Length Violation

When the receive frame length exceeds MAX\_FL bytes the BABR interrupt is generated, and the LG bit in the end of frame RxBD is set. The frame is not truncated unless the frame length exceeds 2047 bytes).

### 24.5.3.3.5 Truncation

When the receive frame length exceeds 2047 bytes, the frame is truncated and the TR bit is set in the RxBD.

## 24.5.4 Full-Duplex Flow Control

Full-duplex flow control allows the user to transmit pause frames and to detect received pause frames.

Upon detection of a pause frame, FEC data frame transmission stops for a given pause duration.

To enable pause frame detection, the FEC must operate in full-duplex mode (FEC\_TCR[FDEN] asserted) and flow control enable (FEC\_RCR[FCE]) must be asserted. The FEC detects a pause frame when the fields of the incoming frame match the pause frame specifications, as shown in [Table 24-6](#). In addition, the receive status associated with the frame indicates that the frame is valid.

**Table 24-6. Pause Frame Field Specification**

48-bit destination address	0x0180_C200_0001 or physical address
48-bit source address	Any
16-bit type	0x8808
16-bit opcode	0x0001
16-bit pause duration	0x0000-0xFFFF

Pause frame detection is performed by the receiver and microcontroller modules. The microcontroller runs an address recognition subroutine to detect the specified pause frame destination address, while the receiver detects the type and opcode pause frame

fields. On detection of a pause frame, FEC\_TCR[GTS] is asserted by the FEC internally. When transmission has paused, the FEC\_EIR[GRA] interrupt is asserted and the pause timer begins to increment.

### NOTE

The pause timer makes use of the transmit backoff timer hardware, which is used for tracking the appropriate collision backoff time in half-duplex mode. The pause timer increments after every slot time, until FEC\_OPD[PAUSE\_DUR] slot times have expired.

On FEC\_OPD[PAUSE\_DUR] expiration, FEC\_TCR[GTS] is cleared allowing FEC data frame transmission to resume.

### NOTE

The receive flow control pause (FEC\_TCR[RFC\_PAUSE]) status bit is asserted while the transmitter is paused due to reception of a pause frame.

To transmit a pause frame, the FEC must operate in full-duplex mode and the user must assert flow control pause (FEC\_TCR[TFC\_PAUSE]). On assertion of transmit flow control pause (FEC\_TCR[TFC\_PAUSE]), the transmitter asserts FEC\_TCR[GTS] internally. When the transmission of data frames stops, the FEC\_EIR[GRA] (graceful stop complete) interrupt asserts. Following FEC\_EIR[GRA] assertion, the pause frame is transmitted. On completion of pause frame transmission, flow control pause (FEC\_TCR[TFC\_PAUSE]) and FEC\_TCR[GTS] are cleared internally.

The user must specify the desired pause duration in the FEC\_OPD register.

### NOTE

When the transmitter is paused due to receiver/microcontroller pause frame detection, transmit flow control pause (FEC\_TCR[TFC\_PAUSE]) still can be asserted and causes the transmission of a single pause frame. In this case, the FEC\_EIR[GRA] interrupt is not asserted.

## 24.5.5 Internal and External Loopback

Both internal and external loopback are supported by the Ethernet controller. In loopback mode, both of the FIFOs are used and the FEC actually operates in a full-duplex fashion. Both internal and external loopback are configured using combinations of the LOOP and

DRT bits in the FEC\_RCR register, the FDEN bit in the FEC\_TCR register and the LBMODE bit in FEC\_MIIGSK\_CFGR register. Furthermore internal loopback has two levels which loops data back in MII domain and RMII domain respectively.

For both internal and external loopback set FDEN = 1 and FEC\_RCR[DRT] = 0.

For the MII domain internal loopback set FEC\_RCR[LOOP] = 1 and FEC\_MIIGSK\_CFGR[LBMODE] = 0. FEC\_TX\_EN and FEC\_TX\_ER will not assert during internal loopback. During the MII domain internal loopback, the transmit/receive data rate is higher than in normal operation because the internal system clock is used by the transmit and receive blocks instead of the clocks from the external transceiver. This will cause an increase in the required system bus bandwidth for transmit and receive data being DMA'd to/from external memory. It may be necessary to pace the frames on the transmit side and/or limit the size of the frames to prevent transmit FIFO underrun and receive FIFO overflow.

For the RMII domain internal loopback, set FEC\_RCR[LOOP] = 0 and FEC\_MIIGSK\_CFGR[LBMODE] = 1. The data is sent from the MII domain into MIIGSK, converted to RMII format by a RMII transmitter inside the MIIGSK, then looped back through a RMII receiver of the MIIGSK, finally gets back to the MII domain.

For external loopback set FEC\_RCR[LOOP] = 0, FEC\_MIIGSK\_CFGR[LBMODE] = 0, and configure the external transceiver for loopback.

## **24.6 Initialization/Application Information**

### **24.6.1 Initialization Sequence**

This section describes which registers are reset due to hardware reset, which are reset by the FEC RISC, and what locations the user must initialize prior to enabling the FEC.

#### **24.6.1.1 Hardware Controlled Initialization**

In the FEC, registers and control logic that generate interrupts are reset by hardware. A hardware reset negates output signals and resets general configuration bits.

Other registers are reset when the FEC\_ECR[ETHER\_EN] bit is cleared, as indicated in [Table 24-7](#). FEC\_ECR[ETHER\_EN] is cleared by a hard reset or can be cleared by software to halt operation. By clearing FEC\_ECR[ETHER\_EN], the configuration control registers such as the FEC\_TCR and FEC\_RCR do not reset, but the entire data path resets.

**Table 24-7. Effect on FEC of Clearing FEC\_ECR[ETHER\_EN]**

Register/Machine	Reset Value
XMIT block	Transmission is aborted (bad CRC appended)
RECV block	Receive activity is aborted
DMA block	All DMA activity is terminated
FEC_RDAR	Cleared
FEC_TDAR	Cleared
Descriptor Controller block	Halt operation
MIIGSK	Interface to transceiver is disabled

### 24.6.1.2 User Initialization (Prior to Asserting FEC\_ECR[ETHER\_EN])

The user must initialize portions of the FEC prior to setting the FEC\_ECR[ETHER\_EN] bit. The exact values depend on the particular application. The order of initializations is not important except those that are explicitly mentioned.

The FEC and FEC FIFO/DMA registers which require initialization are defined in [Table 24-8](#).

**Table 24-8. Registers Requiring Initialization before Setting FEC\_ECR[ETHER\_EN]**

Register	Location (FEC or FIFO/DMA)	Comments
FEC_EIMR	FEC	Requires initialization
FEC_EIR	FEC	Must be cleared by writing 0xFFFF_FFFF
FEC_TFWR	FEC	Optional
FEC_IALR / FEC_IAUR	FEC	-
FEC_GAUR / FEC_GALR	FEC	-
FEC_PALR / FEC_PAUR	FEC	-
FEC_OPD	FEC	Only needed for full-duplex flow control
FEC_RCR	FEC	-
FEC_TCR	FEC	-
FEC_MSCR	FEC	Optional

*Table continues on the next page...*

**Table 24-8. Registers Requiring Initialization before Setting FEC\_ECR[ETHER\_EN]  
(continued)**

Register	Location (FEC or FIFO/DMA)	Comments
FEC_MIIGSK_CFGR	FEC	Must be configured before setting FEC_MIIGSK_ENR[EN]
FEC_MIIGSK_ENR	FEC	EN bit must be set
MIB counters	FEC	Must be cleared
FEC_FRSR	FIFO/DMA	Initialization is optional
FEC_EMRR	FIFO/DMA	-
FEC_ERDSR	FIFO/DMA	-
FEC_ETDSR	FIFO/DMA	-
Transmit Descriptor ring	FIFO/DMA	Must be emptied
Receive Descriptor ring	FIFO/DMA	Must be emptied

### 24.6.1.3 Microcontroller Initialization

In the FEC, the descriptor control RISC initializes some registers after FEC\_ECR[ETHER\_EN] is asserted. After the microcontroller initialization sequence is complete, the hardware is ready for operation.

The microcontroller initialization sequence follows:

1. Initialize BackOff Random Number Seed
2. Activate Receiver
3. Activate Transmitter
4. Clear Transmit FIFO
5. Clear Receive FIFO
6. Initialize Transmit Ring Pointer
7. Initialize Receive Ring Pointer
8. Initialize FIFO Count Register

### 24.6.1.4 User Initialization (after asserting FEC\_ECR[ETHER\_EN])

After asserting FEC\_ECR[ETHER\_EN], the user can set up the buffer/frame descriptors and write to the FEC\_TDAR and FEC\_RDAR. See [Buffer Descriptors](#) for more details.



## 24.6.2 Buffer Descriptors

This section provides a description of the operation of the driver/DMA through the buffer descriptors (BD).

It is followed by a detailed description of the receive and transmit descriptor fields.

### 24.6.2.1 Driver/DMA Operation with Buffer Descriptors

The data for the FEC frames must reside in memory external to the FEC. The data for a frame is placed in one or more buffers. Associated with each buffer is a buffer descriptor (BD) which contains a starting address (pointer), data length, and status/control information (which contains the current state for the buffer). To permit maximum user flexibility, the BDs are also located in external memory and are read in by the FEC DMA engine.

Software "produces" buffers by allocating/initializing memory and initializing buffer descriptors. Setting the RxBD[E] or TxBD[R] bit "produces" the buffer. Software writing to either the FEC\_TDAR or FEC\_RDAR tells the FEC that a buffer has been placed in external memory for the transmit or receive data traffic, respectively. The hardware reads the BDs and "consumes" the buffers after they have been produced. After the data DMA is complete and the buffer descriptor status bits have been written by the DMA engine, the RxBD[E] or TxBD[R] bit is cleared by hardware to signal the buffer has been "consumed." Software can poll the BDs to detect when the buffers have been consumed or can rely on the buffer/frame interrupts. These buffers can then be processed by the driver and returned to the free list.

The FEC\_ECR[ETHER\_EN] signal operates as a reset to the BD/DMA logic. When FEC\_ECR[ETHER\_EN] is cleared the DMA engine BD pointers are reset to point to the starting transmit and receive BDs. The buffer descriptors are not initialized by hardware during reset. At least one transmit and receive buffer descriptor must be initialized by software before the FEC\_ECR[ETHER\_EN] bit is set.

The buffer descriptors operate as two separate rings. FEC\_ERDSR defines the starting address for receive BDs and FEC\_ETDSR defines the starting address for transmit BDs. The last buffer descriptor in each ring is defined by the Wrap (W) bit. When set, W indicates that the next descriptor in the ring is at the location pointed to by FEC\_ERDSR and FEC\_ETDSR for the receive and transmit rings, respectively.

#### NOTE

Buffer descriptor rings must start on a 128-bit boundary.

## 24.6.2.2 Ethernet Transmit Buffer Descriptor (TxBd)

Table 24-9 shows the transmit buffer descriptor format. Table 24-10 describes the TxBd fields.

Status bits for the buffer/frame are not included in the transmit buffer descriptors. Transmit frame status is indicated by individual interrupt bits (error conditions) and in statistic counters in the MIB block. See Message Information Block (MIB) Counters Memory Map, for more details.

**Table 24-9. Transmit Buffer Descriptor (TxBd)**

Offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0004	Tx Data Buffer Pointer A[31:16]															
	Tx Data Buffer Pointer A[15:0]															

**Table 24-10. Transmit Buffer Descriptor Field Definitions**

Offset	Field	Description
0x0000	31 R	Ready. Written by the FEC and the user.  0 The data buffer associated with this BD is not ready for transmission. The user is free to manipulate this BD or its associated data buffer. The FEC clears this bit after the buffer has been transmitted or after an error condition is encountered.  1 The data buffer, which has been prepared for transmission by the user, has not been transmitted or is currently being transmitted. No fields of this BD can be written by the user after this bit is set.
0x0000	30 TO1	Transmit software ownership. This field is reserved for software use. This read/write bit is not modified by hardware, nor does its value affect hardware.
0x0000	29 W	Wrap. Written by user.  0 The next buffer descriptor is found in the consecutive location  1 The next buffer descriptor is found at the location defined in FEC_ETDSR.
0x0000	28 TO2	Transmit software ownership. This field is reserved for use by software. This read/write bit is not modified by hardware, nor does its value affect hardware.
0x0000	27 L	Last in frame. Written by user.  0 The buffer is not the last in the transmit frame.  1 The buffer is the last in the transmit frame.
0x0000	26 TC	Tx CRC. Written by user (only valid if L = 1).  0 End transmission immediately after the last data byte.  1 Transmit the CRC sequence after the last data byte.
0x0000	25 ABC	Append bad CRC. Written by user (only valid if L = 1).  0 No effect  1 Transmit the CRC sequence inverted after the last data byte (regardless of TC value).

*Table continues on the next page...*

**Table 24-10. Transmit Buffer Descriptor Field Definitions (continued)**

Offset	Field	Description
0x0000	15-0 Data Length	Data Length, written by user. Data length is the number of octets the FEC transmits from this BD's data buffer. It is never modified by the FEC. Bits [10:0] are used by the DMA engine, bits[15:11] are ignored.
0x0004	31-0 A	Tx data buffer pointer <sup>1</sup>

1. The transmit buffer pointer, which contains the address of the associated data buffer, must always be divisible by 16. The buffer must reside in memory external to the FEC. This value is never modified by the Ethernet controller.

### 24.6.2.2.1 Driver/DMA Operation with Transmit Buffer Descriptors

Data is presented to the FEC for transmission by arranging it in buffers referenced by the channel's TxBDs. After the software driver has set up the buffers for a frame, it sets up the corresponding BDs. In the TxBD the user initializes the R, W, L, and TC bits and the length (in bytes) in the first longword, and the buffer pointer in the second longword. The last step in setting up the BDs for a transmit frame is to set the R bit in the first BD for the frame. The driver follows that with a write to FEC\_TDAR which triggers the FEC to poll the next BD in the ring.

The Ethernet controller confirms transmission by clearing the ready bit (R bit) when DMA of the buffer is complete.

#### 24.6.2.2.1.1 Transmit Frame in Multiple Buffers

Typically a transmit frame is divided between multiple buffers. For example, it is possible to have an application payload in one buffer, TCP header in a 2nd buffer, IP header in a 3rd buffer, and Ethernet/IEEE 802.3 header in a 4th buffer. The FEC does not prepend the Ethernet header (Destination Address, Source Address, Length/Type field(s)), so this must be provided by the driver in one of the transmit buffers. The FEC or the driver can append the Ethernet CRC to the frame. The driver must set the TC bit in the transmit BD to determine whether the CRC is appended by the FEC or by the driver.

The driver (TxBD software producer) sets up Tx BDs in such a way that a complete transmit frame is given to the hardware at once. If a transmit frame consists of three buffers, first the BD's are initialized with pointer, length and control (W, L, TC, ABC) and then the TxBD[R] bits are set to 1 in *reverse order* (3rd, 2nd, 1st BD) to insure that the complete frame is ready in memory before the DMA begins. If the TxBDs are set up in order, the DMA controller could DMA the first BD before the 2nd was made available, potentially causing a transmit FIFO underrun.

In the FEC, the DMA is notified by the driver that new transmit frame(s) are available by writing to the FEC\_TDAR register. When this register is written to (data value is not significant) the FEC RISC tells the DMA to read the next transmit BD in the ring. After started, the RISC + DMA continues to read and interpret transmit BDs in order and DMA the associated buffers, until a transmit BD is encountered with the R bit = 0. At this point the FEC polls this BD one more time. If the R bit = 0 the second time, then the RISC stops the transmit descriptor read process until software sets up another transmit frame and writes to FEC\_TDAR.

When the DMA of each transmit buffer is complete, the DMA writes back to the BD to clear the R bit, indicating that the hardware consumer is finished with the buffer.

### 24.6.2.3 Ethernet Receive Buffer Descriptor (RxBD)

**Table 24-11. Receive Buffer Descriptor (RxBD)**

Offset	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0x0004	Rx Data Buffer Pointer A[31:16]															
	Rx Data Buffer Pointer A[15:0]															

**Table 24-12. Receive Buffer Descriptor Field Definitions**

Offset	Field	Description
0x0000	31 E	Empty. Written by the FEC (=0) and user (=1).  0 The data buffer associated with this BD has been filled with received data, or data reception has been aborted due to an error condition. The status and length fields have been updated as required.  1 The data buffer associated with this BD is empty, or reception is currently in progress.
0x0000	30 RO1	Receive software ownership.  This field is reserved for use by software. This read/write bit is not modified by hardware, nor does its value affect hardware.
0x0000	29 W	Wrap. Written by user.  0 The next buffer descriptor is found in the consecutive location 1 The next buffer descriptor is found at the location defined in FEC_ERDSR.
0x0000	28 RO2	Receive software ownership.  This field is reserved for use by software. This read/write bit is not modified by hardware, nor does its value affect hardware.
0x0000	27 L	Last in frame. Written by the FEC.  0 The buffer is not the last in a frame. 1 The buffer is the last in a frame.
0x0000	25	Reserved.

*Table continues on the next page...*

**Table 24-12. Receive Buffer Descriptor Field Definitions (continued)**

Offset	Field	Description
0x0000	24 M	Miss. Written by the FEC. This bit is set by the FEC for frames that were accepted in promiscuous mode, but were flagged as a "miss" by the internal address recognition. Thus, while in promiscuous mode, the user can use the M-bit to quickly determine whether the frame was destined to this station. This bit is valid only if the L-bit is set and the PROM bit is set.  0 The frame was received because of an address recognition hit. 1 The frame was received because of promiscuous mode.
0x0000	23 BC	Set if the DA is broadcast (FF:FF:FF:FF:FF:FF).
0x0000	22 MC	Set if the DA is multicast and not BC.
0x0000	21 LG	Rx frame length violation. Written by the FEC. A frame length greater than FEC_RCR[MAX_FL] was recognized. This bit is valid only if the L-bit is set. The receive data is not altered in any way unless the length exceeds 2047 bytes.
0x0000	20 NO	Receive non-octet aligned frame. Written by the FEC. A frame that contained a number of bits not divisible by 8 was received, and the CRC check that occurred at the preceding byte boundary generated an error. This bit is valid only if the L-bit is set. If this bit is set the CR bit is not set.
0x0000	19	Reserved
0x0000	18 CR	Receive CRC error. Written by the FEC. This frame contains a CRC error and is an integral number of octets in length. This bit is valid only if the L-bit is set.
0x0000	17 OV	Overflow. Written by the FEC. A receive FIFO overflow occurred during frame reception. If this bit is set, the other status bits, M, LG, NO, CR, and CL lose their normal meaning and is zero. This bit is valid only if the L-bit is set.
0x0000	16 TR	Set if the receive frame is truncated (frame length > 2047 bytes). If the TR bit is set the frame is discarded and the other error bits ignored as they can be incorrect.
0x0000	15-0 Data Length	Data length. Written by the FEC. Data length is the number of octets written by the FEC into this BD's data buffer if L = 0 (the value is equal to FEC_EMRBR), or the length of the frame including CRC if L = 1. It is written by the FEC once as the BD is closed.
0x0004	31-0	RX data buffer pointer <sup>1</sup>

1. The receive buffer pointer, which contains the address of the associated data buffer, must always be divisible by 16. The buffer must reside in memory external to the FEC. This value is never modified by the Ethernet controller.

### 24.6.2.3.1 Driver/DMA Operation with Receive Buffer Descriptors

Unlike the transmit case, the length of the receive frame is unknown by the driver ahead of time. Therefore the driver must set a variable to define the length of all receive buffers. In the FEC, this variable is written to the FEC\_EMRBR register.

The driver (RxBD software producer) sets up some number of "empty" buffers for the Ethernet by initializing the address field and the E and W bits of the associated receive BDs. The hardware (receive DMA) consumes these buffers by filling them with data as frames are received, and clearing the E bit and writing to the L bit (1 indicates last buffer in frame), the frame status bits (if L = 1) and the length field.

If a receive frame spans multiple receive buffers, the L bit is only set for the last buffer in the frame. For non-last buffers, the length field in the receive BD is written by the DMA (at the same time the E bit is cleared) with the default receive buffer length value. For end of frame buffers, the L=1 bit is set in the receive BD, and information written to the status bits (M, BC, MC, LG, NO, CR, OV, TR). Some of the status bits are error indicators which, if set, indicate the receive frame is discarded and not given to higher layers. The frame status/length information is written into the receive FIFO following the end of the frame (as a single 32-bit word) by the receive logic. The length field for the end of frame buffer is written with the length of the entire frame, not just the length of the last buffer.

For simplicity the driver can assign the default receive buffer length to be large enough to contain an entire frame, keeping in mind that a malfunction on the network or out-of-specification implementation could result in giant frames. Frames of 2 Kbytes (2048 bytes) or larger are truncated by the FEC at 2047 bytes, so software never sees a receive frame larger than 2047 bytes.

As in the transmit case, the FEC polls the receive descriptor ring after the driver sets up receive BDs and writes to the FEC\_RDAR register. As frames are received the FEC fills receive buffers and update the associated BDs, then reads the next BD in the receive descriptor ring. If the FEC reads a receive BD and finds the E bit is cleared, it polls this BD once more. If E is still cleared, then the FEC stops reading receive BDs until the driver writes to FEC\_RDAR.

### NOTE

Whenever the software driver sets an E bit in one or more receive descriptors, the driver follows that with a write to FEC\_RDAR.

## 24.7 Programmable Registers

### 24.7.1 Top Level Block Memory Map

The FEC implementation requires a 1 Kbyte memory space. This space is divided into 2 sections of 512 bytes each. The first is used for control/status registers, and the second contains event/statistics counters held in the MIB block.

[Table 24-13](#) defines the top level memory map. For the base address of a particular block instantiation, see the system memory map.

**Table 24-13. Block Memory Map**

Base Address Offset	Function
0x0000-01FF	Control/Status Registers
0x0200-02FF	MIB Block Counters
0x0300-03FF	Extension Registers (MIIGSK)

## 24.7.2 Message Information Block (MIB) Counters Memory Map

Table 24-14 shows the MIB counters memory map, which defines the locations in the MIB RAM space where hardware maintained counters reside. It is the responsibility of software to poll the counters often enough to ensure that rollover is detected. For example, on a 100 Mbps channel an octets counter could roll over every 5.7 minutes.

These counters fall in the 0x0200-0x03FF address offset range, and are divided into RMON counters and IEEE counters, as follows:

- RMON counters are included which cover the Ethernet statistics counters defined in RFC 1757. In addition to the counters defined in the Ethernet statistics group, a counter is included to count truncated frames as the FEC only supports frame lengths up to 2047 bytes. The RMON counters are implemented independently for transmit and receive to insure accurate network statistics when operating in full-duplex mode.
- IEEE counters are included which support the mandatory and recommended counter packages defined in section 5 of ANSI/IEEE Std. 802.3 (1998 edition). The IEEE basic package objects are supported by the FEC but do not require counters in the MIB block. In addition, some of the recommended package objects which are supported do not require MIB counters. Counters for transmit and receive full-duplex flow control frames are included as well.

**Table 24-14. MIB Counters Memory Map**

Base Address Offset	Mnemonic	Description
0x0200	RMON_T_DROP	Count of frames not counted correctly
0x0204	RMON_T_PACKETS	RMON Tx packet count
0x0208	RMON_T_BC_PKT	RMON Tx broadcast packets
0x020C	RMON_T_MC_PKT	RMON Tx multicast packets
0x0210	RMON_T_CRC_ALIGN	RMON Tx packets w CRC/Align error
0x0214	RMON_T_UNDERSIZE	RMON Tx packets < 64 bytes, good crc
0x0218	RMON_T_OVERSIZE	RMON Tx packets > MAX_FL bytes, good crc
0x021C	RMON_T_FRAG	RMON Tx packets < 64 bytes, bad crc
0x0220	RMON_T_JAB	RMON Tx packets > MAX_FL bytes, bad crc

*Table continues on the next page...*

**Table 24-14. MIB Counters Memory Map (continued)**

Base Address Offset	Mnemonic	Description
0x0224	RMON_T_COL	RMON Tx collision count
0x0228	RMON_T_P64	RMON Tx 64 byte packets
0x022C	RMON_T_P65TO127	RMON Tx 65 to 127 byte packets
0x0230	RMON_T_P128TO255	RMON Tx 128 to 255 byte packets
0x0234	RMON_T_P256TO511	RMON Tx 256 to 511 byte packets
0x0238	RMON_T_P512TO1023	RMON Tx 512 to 1023 byte packets
0x023C	RMON_T_P1024TO2047	RMON Tx 1024 to 2047 byte packets
0x0240	RMON_T_P_GTE2048	RMON Tx packets w > 2048 bytes
0x0244	RMON_T_OCTETS	RMON Tx octets
0x0248	IEEE_T_DROP	Count of frames not counted correctly
0x024C	IEEE_T_FRAME_OK	Frames transmitted OK
0x0250	IEEE_T_1COL	Frames transmitted with single collision
0x0254	IEEE_T_MCOL	Frames transmitted with multiple collisions
0x0258	IEEE_T_DEF	Frames transmitted after deferral delay
0x025C	IEEE_T_LCOL	Frames transmitted with late collision
0x0260	IEEE_T_EXCOL	Frames transmitted with excessive collisions
0x0264	IEEE_T_MACERR	Frames transmitted with Tx FIFO underrun
0x0268	IEEE_T_CSERR	Frames transmitted with carrier sense error
0x026C	IEEE_T_SQE	Frames transmitted with SQE error
0x0270	IEEE_T_FDXFC	Flow control pause frames transmitted
0x0274	IEEE_T_OCTETS_OK	Octet count for frames transmitted w/o error
0x0284	RMON_R_PACKETS	RMON Rx packet count
0x0288	RMON_R_BC_PKT	RMON Rx broadcast packets
0x028C	RMON_R_MC_PKT	RMON Rx multicast packets
0x0290	RMON_R_CRC_ALIGN	RMON Rx packets w CRC/Align error
0x0294	RMON_R_UNDERSIZE	RMON Rx packets < 64 bytes, good crc
0x0298	RMON_R_OVERSIZE	RMON Rx packets > MAX_FL bytes, good crc
0x029C	RMON_R_FRAG	RMON Rx packets < 64 bytes, bad crc
0x02A0	RMON_R_JAB	RMON Rx packets > MAX_FL bytes, bad crc
0x02A4	RMON_R_RESVD_0	-
0x02A8	RMON_R_P64	RMON Rx 64 byte packets
0x02AC	RMON_R_P65TO127	RMON Rx 65 to 127 byte packets
0x02B0	RMON_R_P128TO255	RMON Rx 128 to 255 byte packets
0x02B4	RMON_R_P256TO511	RMON Rx 256 to 511 byte packets
0x02B8	RMON_R_P512TO1023	RMON Rx 512 to 1023 byte packets
0x02BC	RMON_R_P1024TO2047	RMON Rx 1024 to 2047 byte packets
0x02C0	RMON_R_P_GTE2048	RMON Rx packets w > 2048 bytes
0x02C4	RMON_R_OCTETS	RMON Rx octets
0x02C8	IEEE_R_DROP	Count of frames not counted correctly

*Table continues on the next page...*



**Table 24-14. MIB Counters Memory Map (continued)**

Base Address Offset	Mnemonic	Description
0x02CC	IEEE_R_FRAME_OK	Frames received OK
0x02D0	IEEE_R_CRC	Frames received with CRC error
0x02D4	IEEE_R_ALIGN	Frames received with alignment error
0x02D8	IEEE_R_MACERR	Receive FIFO overflow count
0x02DC	IEEE_R_FDXFC	Flow control pause frames received
0x02E0	IEEE_R_OCTETS_OK	Octet count for frames received w/o error

### 24.7.3 MIIGSK Registers Memory Map

**Table 24-15. MIIGSK Registers Memory Map**

Offset	Mnemonic	Description
0x0300	FEC_MIIGSK_CFGR	MIIGSK Configuration Register
0x0308	FEC_MIIGSK_ENR	MIIGSK Enable Register

## 24.8 FEC Memory Map/Register Definition

This section includes the FEC memory map and detailed descriptions of all the registers, followed by a description of the buffers.

The FEC is programmed by a combination of control and status registers (FEC\_CSRs) and buffer descriptors. The FEC\_CSRs are used for mode control and to extract global status information. The descriptors are used to pass data buffers and related buffer information between the hardware and software.

The FEC implementation requires a 1 Kbyte memory space. This space is divided into 2 sections of 512 bytes each. The first is used for control/status registers, and the second contains event/statistics counters held in the MIB block.

Table defines the top level memory map. For the base address of a particular block instantiation, see the system memory map.

**Table 24-17. Block Memory Map**

Base Address Offset	Function
---------------------	----------

*Table continues on the next page...*

**Table 24-17. Block Memory Map (continued)**

0x0000-01FF	Control/Status Registers
0x0200-02FF	MIB Block Counters
0x0300-03FF	Extension Registers (MIIGSK)

The following table shows the FEC register memory map. For the base address of a particular block instantiation, see the system memory map.

**FEC memory map**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/page</b>
218_8004	Ethernet interrupt event register (FEC_EIR)	32	R/W	0000_0000h	<a href="#">24.8.1/1079</a>
218_8008	Ethernet interrupt mask register (FEC_EIMR)	32	R/W	0000_0000h	<a href="#">24.8.2/1081</a>
218_8010	Receive descriptor active register (FEC_RDAR)	32	R/W	0000_0000h	<a href="#">24.8.3/1084</a>
218_8014	Transmit descriptor active register (FEC_TDAR)	32	R/W	0000_0000h	<a href="#">24.8.4/1085</a>
218_8024	Ethernet control register (FEC_ECR)	32	R/W	F000_0000h	<a href="#">24.8.5/1086</a>
218_8040	MII management frame register (FEC_MMFR)	32	R/W	Unaffected	<a href="#">24.8.6/1087</a>
218_8044	MII speed control register (FEC_MSCR)	32	R/W	0000_0000h	<a href="#">24.8.7/1088</a>
218_8064	MIB control register (FEC_MIBC)	32	R/W	C000_0000h	<a href="#">24.8.8/1090</a>
218_8084	Receive control register (FEC_RCR)	32	R/W	05EE_0001h	<a href="#">24.8.9/1091</a>
218_80C4	Transmit control register (FEC_TCR)	32	R/W	0000_0000h	<a href="#">24.8.10/1093</a>
218_80E4	Physical address low register (FEC_PALR)	32	R/W	0000_0000h	<a href="#">24.8.11/1095</a>
218_80E8	Physical address upper register (FEC_PAUR)	32	R/W	0000_8808h	<a href="#">24.8.12/1095</a>
218_80EC	Opcode and pause duration register (FEC_OPDR)	32	R/W	0001_0000h	<a href="#">24.8.13/1096</a>
218_8118	Descriptor individual address upper register (FEC_IAUR)	32	R/W	0000_0000h	<a href="#">24.8.14/1097</a>
218_811C	Descriptor individual address lower register (FEC_IALR)	32	R/W	0000_0000h	<a href="#">24.8.15/1097</a>
218_8120	Descriptor group address upper register (FEC_GAUR)	32	R/W	0000_0000h	<a href="#">24.8.16/1098</a>
218_8124	Descriptor group address lower register (FEC_GALR)	32	R/W	0000_0000h	<a href="#">24.8.17/1098</a>
218_8144	Transmit FIFO watermark register (FEC_TFWR)	32	R/W	0000_0000h	<a href="#">24.8.18/1099</a>
218_814C	FIFO receive bound register (FEC_FRBR)	32	R	0000_0600h	<a href="#">24.8.19/1100</a>
218_8150	FIFO receive FIFO start registers (FEC_FRSR)	32	R/W	0000_0500h	<a href="#">24.8.20/1100</a>
218_8180	Receive buffer descriptor ring start register (FEC_ERDSR)	32	R/W	Unaffected	<a href="#">24.8.21/1101</a>

*Table continues on the next page...*

**FEC memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
218_8184	Transmit buffer descriptor ring start register (FEC_ETDSR)	32	R/W	0000_0000h	<a href="#">24.8.22/1102</a>
218_8188	Maximum receive buffer size register (FEC_EMRR)	32	R/W	0000_0000h	<a href="#">24.8.23/1102</a>

**24.8.1 Ethernet interrupt event register (FEC\_EIR)**

The FEC\_EIR bit assignments are shown below. When an event occurs that sets a bit in the FEC\_EIR, an interrupt is generated if the corresponding bit in the interrupt mask register (FEC\_EIMR) is also set. The bit in the FEC\_EIR is cleared if a one is written to that bit position; writing zero has no effect. This register is cleared upon hardware reset.

Interrupts can be divided into three classes as follows:

- Interrupts that occur in normal operation: these include GRA, TXF, TXB, RXF, RXB, and MII.
- Interrupts that result from errors or problems detected in the network or transceiver: these include HBERR, BABR, BABT, LC, and RL.
- Interrupts that result from internal errors are EBERR and UN.

Some of the error interrupts are independently counted in the MIB block counters. The correspondence between interrupts and counters is shown in the following table. Software can choose to mask off the interrupts because these errors are visible to network management through the MIB counters.

**Table 24-19. Error Interrupts and Block Counters**

Interrupt	Counter(s)
HBERR	IEEE_T_SQE
BABR	RMON_R_OVERSIZE (good CRC), RMON_R_JAB (bad CRC)
BABT	RMON_T_OVERSIZE (good CRC), RMON_T_JAB (bad CRC)
LATE_COL	IEEE_T_LCOL
COL_RETRY_LIM	IEEE_T_EXCOL
XFIFO_UN	IEEE_T_MACERR

## FEC Memory Map/Register Definition

Address: 218\_8000h base + 4h offset = 218\_8004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HBERR	BABR	BABT	GRA	TXF	TXB	RXF	RXB	MII	EBERR	LC	RL	UN	Reserved		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### FEC\_EIR field descriptions

Field	Description
31 HBERR	Heartbeat error. This interrupt indicates that HBC is set in the FEC_TCR register and that the COL input was not asserted within the Heartbeat window following a transmission.
30 BABR	Babbling receive error. This bit indicates a frame was received with length in excess of FEC_RCR[MAX_FL] bytes.
29 BABT	Babbling transmit error. This bit indicates that the transmitted frame length has exceeded FEC_RCR[MAX_FL] bytes. This condition is usually caused by a frame that is too long being placed into the transmit data buffer(s). Truncation does not occur.
28 GRA	Graceful stop complete. This interrupt is asserted for one of three reasons. Graceful stop means that the transmitter is put into a pause state after completion of the frame currently being transmitted. 1) A graceful stop, which was initiated by the setting of the FEC_TCR[GTS] bit is now complete. 2) A graceful stop, which was initiated by the setting of the FEC_TCR[TFC_PAUSE] bit is now complete. 3) A graceful stop, which was initiated by the reception of a valid full-duplex flow control "pause" frame is now complete. See the "Full-Duplex Flow Control" section of the Functional Description chapter.
27 TXF	Transmit frame interrupt. This bit indicates that a frame has been transmitted and that the last corresponding buffer descriptor has been updated.
26 TXB	Transmit buffer interrupt. This bit indicates that a transmit buffer descriptor has been updated.
25 RXF	Receive frame interrupt. This bit indicates that a frame has been received and that the last corresponding buffer descriptor has been updated.
24 RXB	Receive buffer interrupt. This bit indicates that a receive buffer descriptor has been updated that was not the last in the frame.
23 MII	MII interrupt. This bit indicates that the MII has completed the data transfer requested.
22 EBERR	Ethernet bus error. This bit indicates that a system bus error occurred when a DMA transaction was underway. When the EBERR bit is set, FEC_ECR[ETHER_EN] is cleared, halting frame processing by the FEC_. When this occurs software needs to insure that the FIFO controller and DMA are also soft reset.
21 LC	Late collision. This bit indicates that a collision occurred beyond the collision window (slot time) in half-duplex mode. The frame is truncated with a bad CRC and the remainder of the frame is discarded.
20 RL	Collision retry limit. This bit indicates that a collision occurred on each of 16 successive attempts to transmit the frame. The frame is discarded without being transmitted and transmission of the next frame commences. Can only occur in half-duplex mode.

Table continues on the next page...

**FEC\_EIR field descriptions (continued)**

Field	Description
19 UN	Transmit FIFO underrun. This bit indicates that the transmit FIFO became empty before the complete frame was transmitted. A bad CRC is appended to the frame fragment and the remainder of the frame is discarded.
18–0 -	This field is reserved. Reserved, read as 0

**24.8.2 Ethernet interrupt mask register (FEC\_EIMR)**

The FEC\_EIMR controls which of the interrupt events flagged in the FEC\_EIR are allowed to generate actual interrupts. If the corresponding bits in both the FEC\_EIR and FEC\_EIMR are set, the interrupt is signaled to the ARM platform. The interrupt signal remains asserted until a 1 is written to the FEC\_EIR bit (write 1 to clear) or a 0 is written to the FEC\_EIMR bit. This register is cleared upon a hardware reset.

The FEC\_EIMR bit assignments are the same as for the FEC\_EIR.

Address: 218\_8000h base + 8h offset = 218\_8008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	HBERR	BABR	BABT	GRA	TXF	TXB	RXF	RXB	MII	EBERR	LC	RL	UN	0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**FEC\_EIMR field descriptions**

Field	Description
31 HBERR	Interrupt mask. This bit corresponds to an interrupt source defined by the FEC_EIR register. This FEC_EIMR bit determines whether an interrupt will be generated when the interrupt condition is fulfilled. At every system clock, the FEC_EIR samples the signals generated from the interrupting sources. The corresponding FEC_EIR bit reflects the state of the interrupt signal even if the corresponding FEC_EIMR bit is cleared.  0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.
30 BABR	Interrupt mask. This bit corresponds to an interrupt source defined by the FEC_EIR register. This FEC_EIMR bit determines whether an interrupt will be generated when the interrupt condition is fulfilled. At

*Table continues on the next page...*

**FEC\_EIMR field descriptions (continued)**

Field	Description
	<p>every system clock, the FEC_EIR samples the signals generated from the interrupting sources. The corresponding FEC_EIR bit reflects the state of the interrupt signal even if the corresponding FEC_EIMR bit is cleared.</p> <p>0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.</p>
29 BABT	<p>Interrupt mask. This bit corresponds to an interrupt source defined by the FEC_EIR register. This FEC_EIMR bit determines whether an interrupt will be generated when the interrupt condition is fulfilled. At every system clock, the FEC_EIR samples the signals generated from the interrupting sources. The corresponding FEC_EIR bit reflects the state of the interrupt signal even if the corresponding FEC_EIMR bit is cleared.</p> <p>0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.</p>
28 GRA	<p>Interrupt mask. This bit corresponds to an interrupt source defined by the FEC_EIR register. This FEC_EIMR bit determines whether an interrupt will be generated when the interrupt condition is fulfilled. At every system clock, the FEC_EIR samples the signals generated from the interrupting sources. The corresponding FEC_EIR bit reflects the state of the interrupt signal even if the corresponding FEC_EIMR bit is cleared.</p> <p>0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.</p>
27 TXF	<p>Interrupt mask. This bit corresponds to an interrupt source defined by the FEC_EIR register. This FEC_EIMR bit determines whether an interrupt will be generated when the interrupt condition is fulfilled. At every system clock, the FEC_EIR samples the signals generated from the interrupting sources. The corresponding FEC_EIR bit reflects the state of the interrupt signal even if the corresponding FEC_EIMR bit is cleared.</p> <p>0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.</p>
26 TXB	<p>Interrupt mask. This bit corresponds to an interrupt source defined by the FEC_EIR register. This FEC_EIMR bit determines whether an interrupt will be generated when the interrupt condition is fulfilled. At every system clock, the FEC_EIR samples the signals generated from the interrupting sources. The corresponding FEC_EIR bit reflects the state of the interrupt signal even if the corresponding FEC_EIMR bit is cleared.</p> <p>0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.</p>
25 RXF	<p>Interrupt mask. This bit corresponds to an interrupt source defined by the FEC_EIR register. This FEC_EIMR bit determines whether an interrupt will be generated when the interrupt condition is fulfilled. At every system clock, the FEC_EIR samples the signals generated from the interrupting sources. The corresponding FEC_EIR bit reflects the state of the interrupt signal even if the corresponding FEC_EIMR bit is cleared.</p> <p>0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.</p>
24 RXB	<p>Interrupt mask. This bit corresponds to an interrupt source defined by the FEC_EIR register. This FEC_EIMR bit determines whether an interrupt will be generated when the interrupt condition is fulfilled. At every system clock, the FEC_EIR samples the signals generated from the interrupting sources. The corresponding FEC_EIR bit reflects the state of the interrupt signal even if the corresponding FEC_EIMR bit is cleared.</p>

*Table continues on the next page...*

**FEC\_EIMR field descriptions (continued)**

<b>Field</b>	<b>Description</b>
	0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.
23 MII	Interrupt mask. This bit corresponds to an interrupt source defined by the FEC_EIR register. This FEC_EIMR bit determines whether an interrupt will be generated when the interrupt condition is fulfilled. At every system clock, the FEC_EIR samples the signals generated from the interrupting sources. The corresponding FEC_EIR bit reflects the state of the interrupt signal even if the corresponding FEC_EIMR bit is cleared.  0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.
22 EBERR	Interrupt mask. This bit corresponds to an interrupt source defined by the FEC_EIR register. This FEC_EIMR bit determines whether an interrupt will be generated when the interrupt condition is fulfilled. At every system clock, the FEC_EIR samples the signals generated from the interrupting sources. The corresponding FEC_EIR bit reflects the state of the interrupt signal even if the corresponding FEC_EIMR bit is cleared.  0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.
21 LC	Interrupt mask. This bit corresponds to an interrupt source defined by the FEC_EIR register. This FEC_EIMR bit determines whether an interrupt will be generated when the interrupt condition is fulfilled. At every system clock, the FEC_EIR samples the signals generated from the interrupting sources. The corresponding FEC_EIR bit reflects the state of the interrupt signal even if the corresponding FEC_EIMR bit is cleared.  0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.
20 RL	Interrupt mask. This bit corresponds to an interrupt source defined by the FEC_EIR register. This FEC_EIMR bit determines whether an interrupt will be generated when the interrupt condition is fulfilled. At every system clock, the FEC_EIR samples the signals generated from the interrupting sources. The corresponding FEC_EIR bit reflects the state of the interrupt signal even if the corresponding FEC_EIMR bit is cleared.  0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.
19 UN	Interrupt mask. This bit corresponds to an interrupt source defined by the FEC_EIR register. This FEC_EIMR bit determines whether an interrupt will be generated when the interrupt condition is fulfilled. At every system clock, the FEC_EIR samples the signals generated from the interrupting sources. The corresponding FEC_EIR bit reflects the state of the interrupt signal even if the corresponding FEC_EIMR bit is cleared.  0 The corresponding interrupt source is masked. 1 The corresponding interrupt source is not masked.
18–0 Reserved	This read-only field is reserved and always has the value 0.

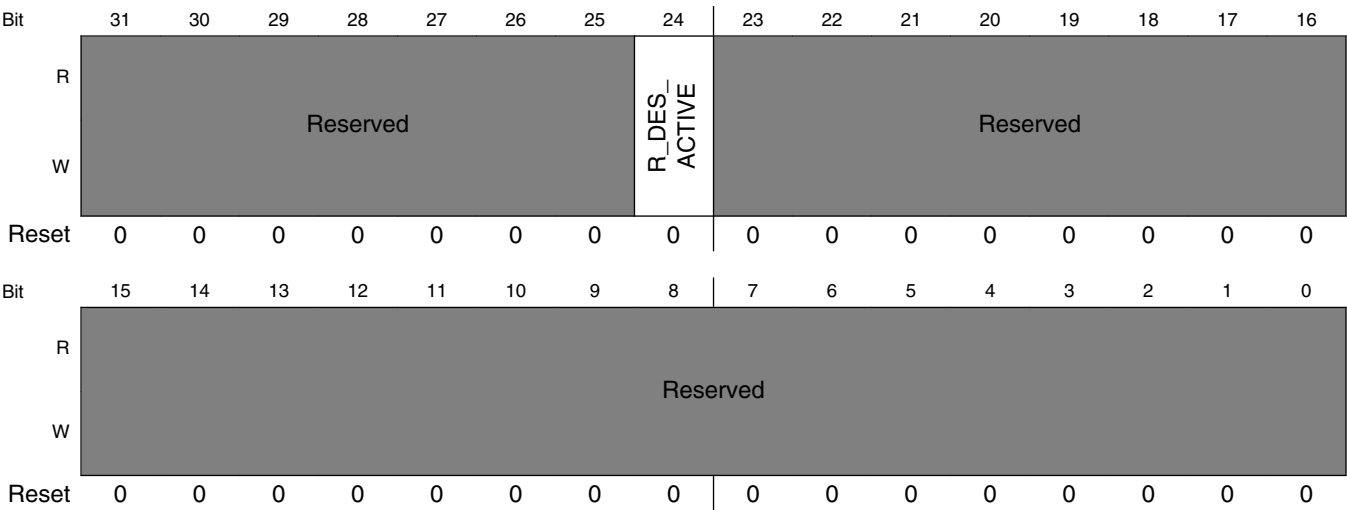
### 24.8.3 Receive descriptor active register (FEC\_RDAR)

FEC\_RDAR is a user-writeable command register which indicates that the receive descriptor ring has been updated, and that empty receive buffers have been produced by the driver with the empty bit in RxBD set.

The FEC\_RDAR[R\_DES\_ACTIVE] bit is set whenever the register is written, independent of the data actually written by the user. When it is set, the FEC polls the receive descriptor ring and processes receive frames (provided FEC\_ECR[ETHER\_EN] is also set to 1). After the FEC finds a receive descriptor whose empty bit is not set, the FEC clears the FEC\_RDAR[R\_DES\_ACTIVE] bit and ceases the polling in the receive descriptor ring until the user sets the bit again to sign the availability of additional descriptors in the receive descriptor ring.

The FEC\_RDAR is cleared when FEC\_ECR[ETHER\_EN] is cleared or system is reset.

Address: 218\_8000h base + 10h offset = 218\_8010h



FEC\_RDAR field descriptions

Field	Description
31–25 -	This field is reserved. Reserved, read as 0
24 R_DES_ACTIVE	Set to one when this register is written, regardless of the value written. Cleared by the FEC device when the FEC finds a receive descriptor whose empty bit is not set. Also cleared when FEC_ECR[ETHER_EN] is cleared.
23–0 -	This field is reserved. Reserved, read as 0



## 24.8.4 Transmit descriptor active register (FEC\_TDAR)

The FEC\_TDAR is a command register, which is written by the user, to indicate that the transmit descriptor ring has been updated (transmit buffers have been produced by the driver with the ready bit set in the buffer descriptor).

Whenever the register is written, the FEC\_TDAR[X\_DES\_ACTIVE] bit is set to 1, independent of the data actually written by the user. When set, the FEC polls the transmit descriptor ring and process transmit frames (provided FEC\_ECR[ETHER\_EN] is also set to 1). After the FEC finds a transmit descriptor whose ready bit is not set, then the FEC clears the FEC\_TDAR[X\_DES\_ACTIVE] bit and ceases transmit descriptor ring polling until the user sets the bit again, the transmit descriptor ring.

The FEC\_TDAR is cleared, when FEC\_ECR[ETHER\_EN] is cleared, or when FEC\_ECR[RESET] is set to 1 or system is reset.

Address: 218\_8000h base + 14h offset = 218\_8014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved							X_DES_ACTIVE	Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

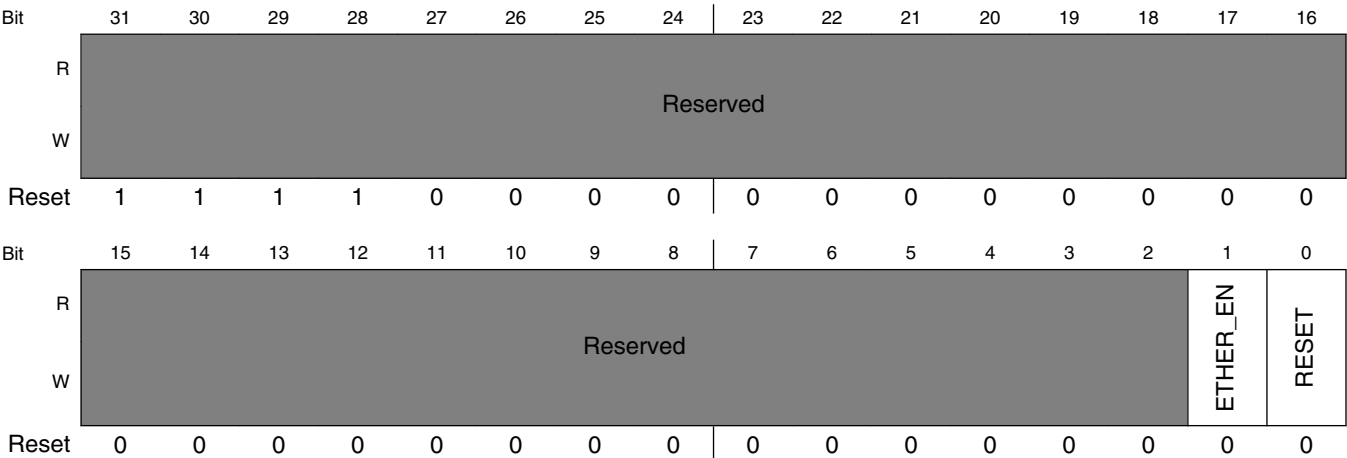
**FEC\_TDAR field descriptions**

Field	Description
31–25 -	This field is reserved. Reserved, read as 0
24 X_DES_ACTIVE	Set to one when this register is written, regardless of the value written. Cleared by the FEC device when the FEC finds a transmit descriptor whose ready bit is not set. Also cleared when FEC_ECR[ETHER_EN] is cleared.
23–0 -	This field is reserved. Reserved, read as 0

24.8.5 Ethernet control register (FEC\_ECR)

The ECR is used to enable/disable the FEC\_ECR which is a read/write user register, though both fields in this register can also be altered by hardware.

Address: 218\_8000h base + 24h offset = 218\_8024h



FEC\_ECR field descriptions

Field	Description
31–2 -	This field is reserved. Reserved.
1 ETHER_EN	When this bit is set, the FEC is enabled, and reception and transmission are possible. When this bit is cleared, reception is immediately stopped and transmission is stopped after a bad CRC is appended to any currently transmitted frame. The buffer descriptor(s) for an aborted transmit frame are not updated after clearing this bit. When ETHER_EN is cleared, the DMA, buffer descriptor, and FIFO control logic are reset, including the buffer descriptor and FIFO pointers. The ETHER_EN bit is altered by hardware under the following conditions:  FEC_ECR[RESET] is set by software, in which case ETHER_EN is cleared  an error condition causes the FEC_EIR[EBERR] bit to set, in which case ETHER_EN is cleared
0 RESET	When this bit is set, the equivalent of a hardware reset is performed but it is local to the FEC, ETHER_EN is cleared and all other FEC registers take their reset values. Also, any transmission/reception currently in progress is abruptly aborted. This bit is automatically cleared by hardware during the reset sequence. The reset sequence takes approximately 8 system clock cycles after RESET is written with a 1.

### 24.8.6 MII management frame register (FEC\_MMFR)

The FEC\_MMFR is used to communicate with the attached MII compatible PHY device(s) by providing read/write access to their MII registers. Performing a write to FEC\_MMFR causes a management frame to be generated unless the MII-SPEED field of the FEC\_MSCR has been set to 0. MII frame is generated with the data previously written to FEC\_MMFR. This allows FEC\_MMFR and FEC\_MSCR to be programmed in any order if the MII-SPEED field of FEC\_MSCR is zero (for further details see [MII speed control register \(FEC\\_MSCR\)](#) ).

The FEC\_MMFR does not reset to a definite value.

To perform a read or write operation on the MII Management Interface, the FEC\_MMFR must be written by the user. To generate a valid read or write management frame, the ST field must be written with a 01 pattern, and the TA field must be written with a 10 pattern. If other patterns are written to these fields, a frame is generated but does not comply with the IEEE 802.3 MII definition.

To generate an IEEE 802.3-compliant MII management interface write frame (write to a PHY register), the user must write {01 01 PHYAD REGAD 10 DATA} to the bit fields of the FEC\_MMFR, as shown in [MII management frame register \(FEC\\_MMFR\)](#) , causes the control logic to shift out the data in the FEC\_MMFR following a preamble generated by the control state machine. During this time, the contents of the FEC\_MMFR is altering as the contents are serially shifted so the value is unpredictable if it is read by the user. After the write management frame operation has completed, the MII interrupt is generated. At this time, the contents of the FEC\_MMFR matches to the original value written.

To generate an MII management interface read frame (read a PHY register) the user must write {01 10 PHYAD REGAD 10 XXXX} to the bit fields of the FEC\_MMFR shown in [MII management frame register \(FEC\\_MMFR\)](#) (the contents of the 4-bit DATA field are arbitrary). This causes the control logic to shift out the data in the FEC\_MMFR following a preamble generated by the control state machine. During this time, the contents of the FEC\_MMFR is altering as the contents are serially shifted to the register value, it is unpredictable if it is read by the user. After the read management frame operation has completed, the MII interrupt is generated. At this time the contents of the FEC\_MMFR matches to the original value written except for the DATA field whose contents have been replaced by the value read from the PHY register.

If the FEC\_MMFR is written during the progress of frame greneration, the frame contents is altered. Thus software should use the MII interrupt to avoid writing to the FEC\_MMFR while frame generation is in progress.

## FEC Memory Map/Register Definition

Address: 218\_8000h base + 40h offset = 218\_8040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W	ST		OP		PA				RA				TA		DATA																	
Reset	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	

\* Notes:

- u = Unaffected by reset.

## FEC\_MMFR field descriptions

Field	Description
31–30 ST	Start of frame delimiter. These bits must be programmed to 01 for a valid MII management frame.
29–28 OP	Operation code. This field must be programmed to 10 (read) or 01 (write) to generate a valid MII management frame. A value of 11 produces "read" frame operation while a value of 00 produces "write" frame operation, but these frames is not MII compliant.
27–23 PA	PHY address. This field specifies one of up to 32 attached PHY devices.
22–18 RA	Register address. This field specifies one of up to 32 registers within the specified PHY device.
17–16 TA	Turn around. This field must be programmed to 10 to generate a valid MII management frame.
15–0 DATA	Management frame data. This is the field for data to be written to or read from the PHY register.

## 24.8.7 MII speed control register (FEC\_MSCR)

The FEC\_MSCR provides control of the frequency of the MII clock (FEC\_MDC signal), and allows a preamble drop on the MII management frame .

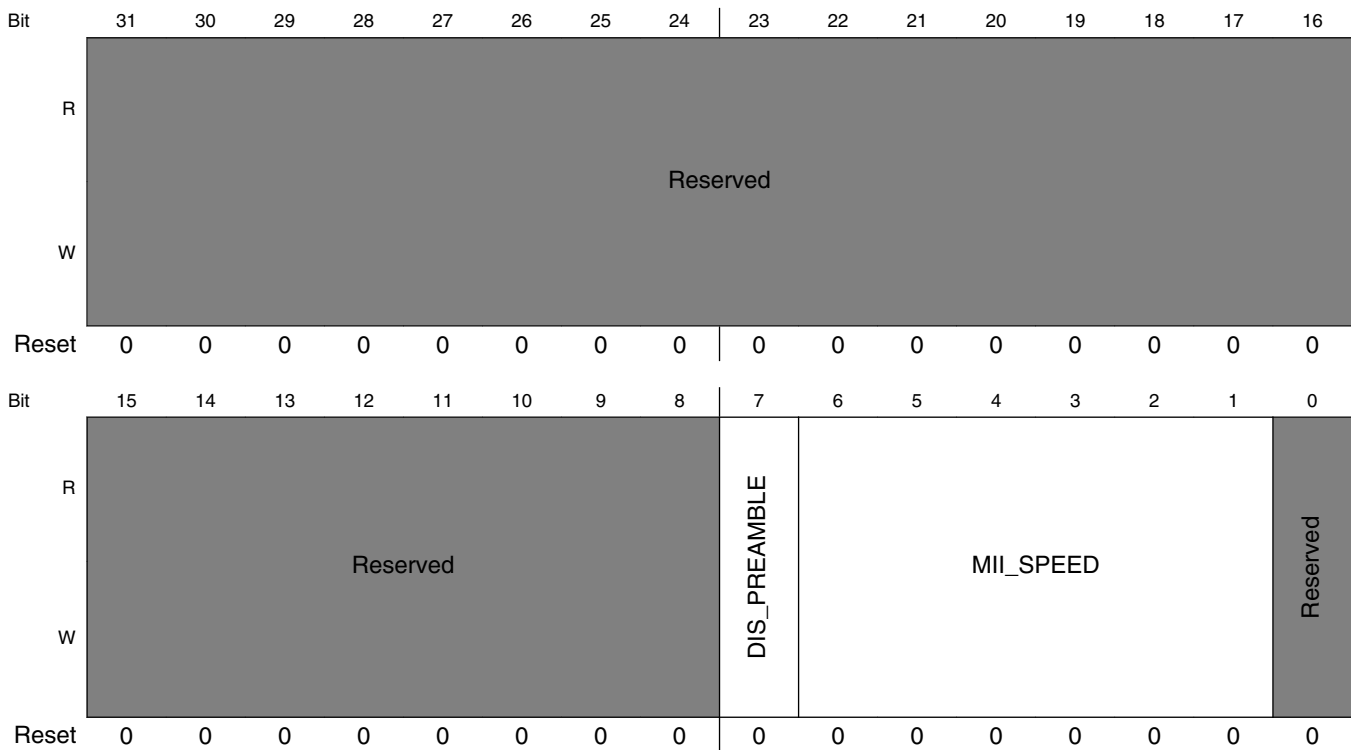
The MII\_SPEED field must be programmed with a value to generate an FEC\_MDC frequency which is less than or equal to 2.5 MHz so that it is compliant with the IEEE 802.3 MII specification. The MII\_SPEED must be set to a non-zero value in order to generate a read or write management frame. After the management frame is completed, the FEC\_MSCR can optionally be set to zero in order to turn off the FEC\_MDC. The FEC\_MDC frequency has a 50% duty cycle except when MII\_SPEED is changed during operation (change takes effect following either a rising or falling edge of FEC\_MDC).

The FEC\_MDC frequency depends on both the system clock frequency and the MII\_SPEED register. If the system clock is 25 MHz, programming the MII\_SPEED register to 0x0000\_0005 results in an FEC\_MDC frequency of  $25 \text{ MHz} * 1/10 = 2.5 \text{ MHz}$ . A table showing optimum values for MII\_SPEED at different system clock frequencies is provided below.

**Table 24-26. Programming Examples for FEC\_MSCR**

System Clock Frequency	MII_SPEED (field in reg)	FEC_MDC frequency
25 MHz	0x5	2.5 MHz
33 MHz	0x7	2.36 MHz
40 MHz	0x8	2.5 MHz
50 MHz	0xA	2.5 MHz
66 MHz	0xD	2.54 MHz

Address: 218\_8000h base + 44h offset = 218\_8044h

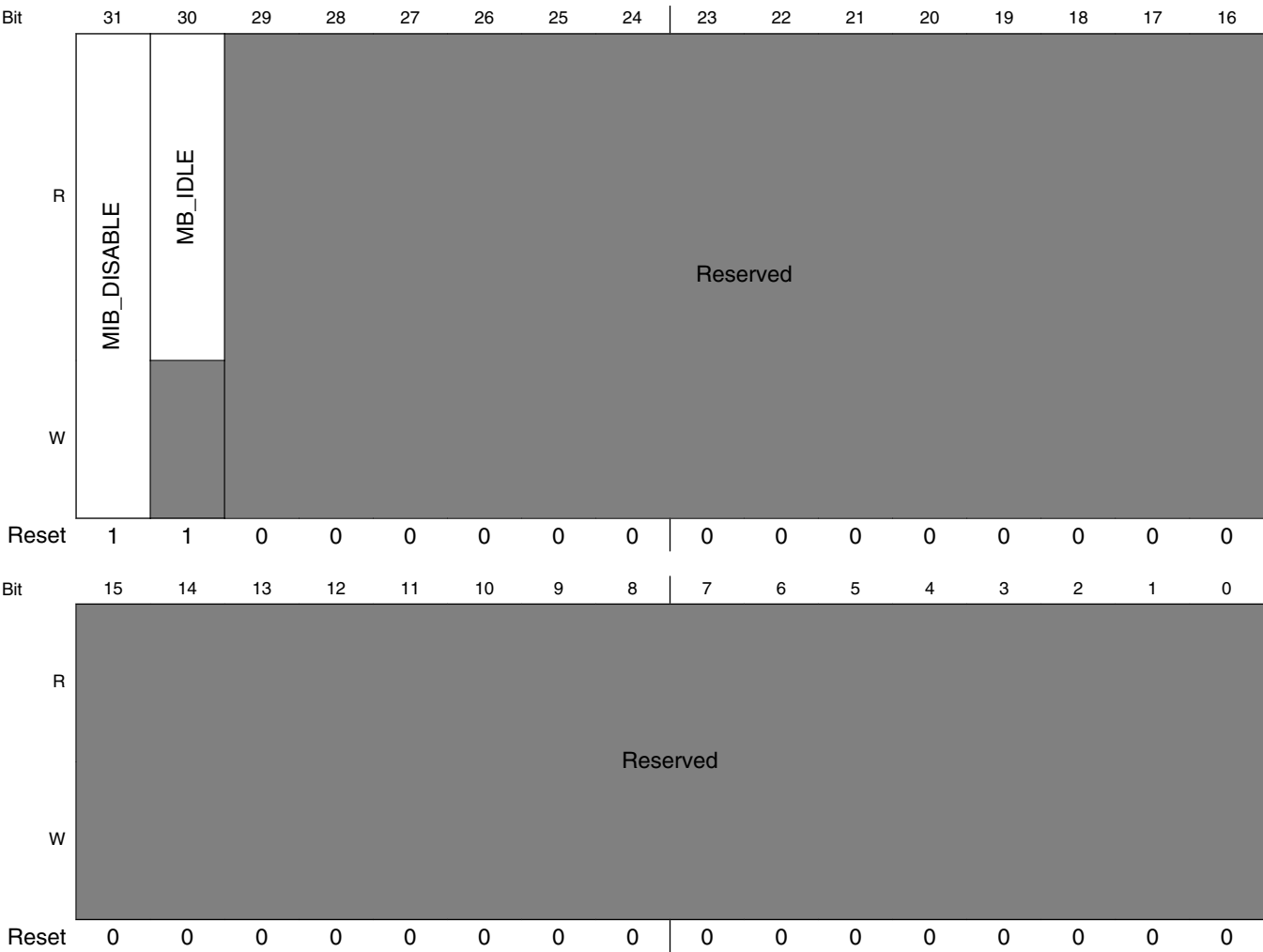
**FEC\_MSCR field descriptions**

Field	Description
31–8 -	This field is reserved. Reserved, read as 0
7 DIS_PREAMBLE	Asserting this bit causes preamble (0xFFFF_FFFF) not to be prepended to the MII management frame. The MII standard allows the preamble to be dropped if the attached PHY device(s) does not require it.
6–1 MII_SPEED	MII_SPEED controls the frequency of the MII management interface clock (FEC_MDC) relative to the system clock. A value of 0 in this field "turns off" the FEC_MDC and leave it in low voltage state. Any non-zero value results in the FEC_MDC frequency of 1/(MII_SPEED*2) of the system clock frequency.
0 -	This field is reserved. Reserved, read as 0

24.8.8 MIB control register (FEC\_MIBC)

The MIB control register is a read/write register which is used to provide control of and to observe the state of the Message Information Block (MIB). This register is accessed by user software if there is a need to disable the MIB operation. For example, the user wants to clear all MIB counters in RAM. the user disables the MIB, then clears all the MIB RAM locations, then enables the MIB. The reset value of MIB\_DISABLE bit is 1. See [Table 24-14](#) for the locations of the MIB counters.

Address: 218\_8000h base + 64h offset = 218\_8064h



**FEC\_MIBC field descriptions**

Field	Description
31 MIB_DISABLE	A read/write control bit. If set, the MIB logic halts and does not update any MIB counters.
30 MB_IDLE	A read-only status bit. If set, the MIB block is not currently updating any MIB counters.
29–0 -	This field is reserved. Reserved.

**24.8.9 Receive control register (FEC\_RCR)**

The FEC\_RCR is programmed by the user, and controls the operational mode of the receive block. It can only be written when FEC\_ECR[ETHER\_EN] = 0 (that is, during initialization).

Address: 218\_8000h base + 84h offset = 218\_8084h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved					MAX_FL										
W																
Reset	0	0	0	0	0	1	0	1	1	1	1	0	1	1	1	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										FCE		BC_REJ	PROM	MII_MODE	DRT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

**FEC\_RCR field descriptions**

Field	Description
31–27 -	This field is reserved. Reserved, read as 0
26–16 MAX_FL	Maximum frame length. Resets to decimal 1518. Length is measured from DA and up to the CRC at the end of the frame. Transmit frames longer than MAX_FL causes the BABT interrupt to occur. Receive Frames longer than MAX_FL causes the BABR interrupt to occur and sets the LG bit in the last frame receive buffer descriptor. The recommended default value to be programmed by the user is 1518 or 1522 (if VLAN Tags are supported).
15–6 -	This field is reserved. Reserved, read as 0
5 FCE	Flow control enable. When FCE is set to 1, the receiver detects pause frames. Upon pause frame detection, the transmitter stops transmitting data frames for a given duration.

Table continues on the next page...

**FEC\_RCR field descriptions (continued)**

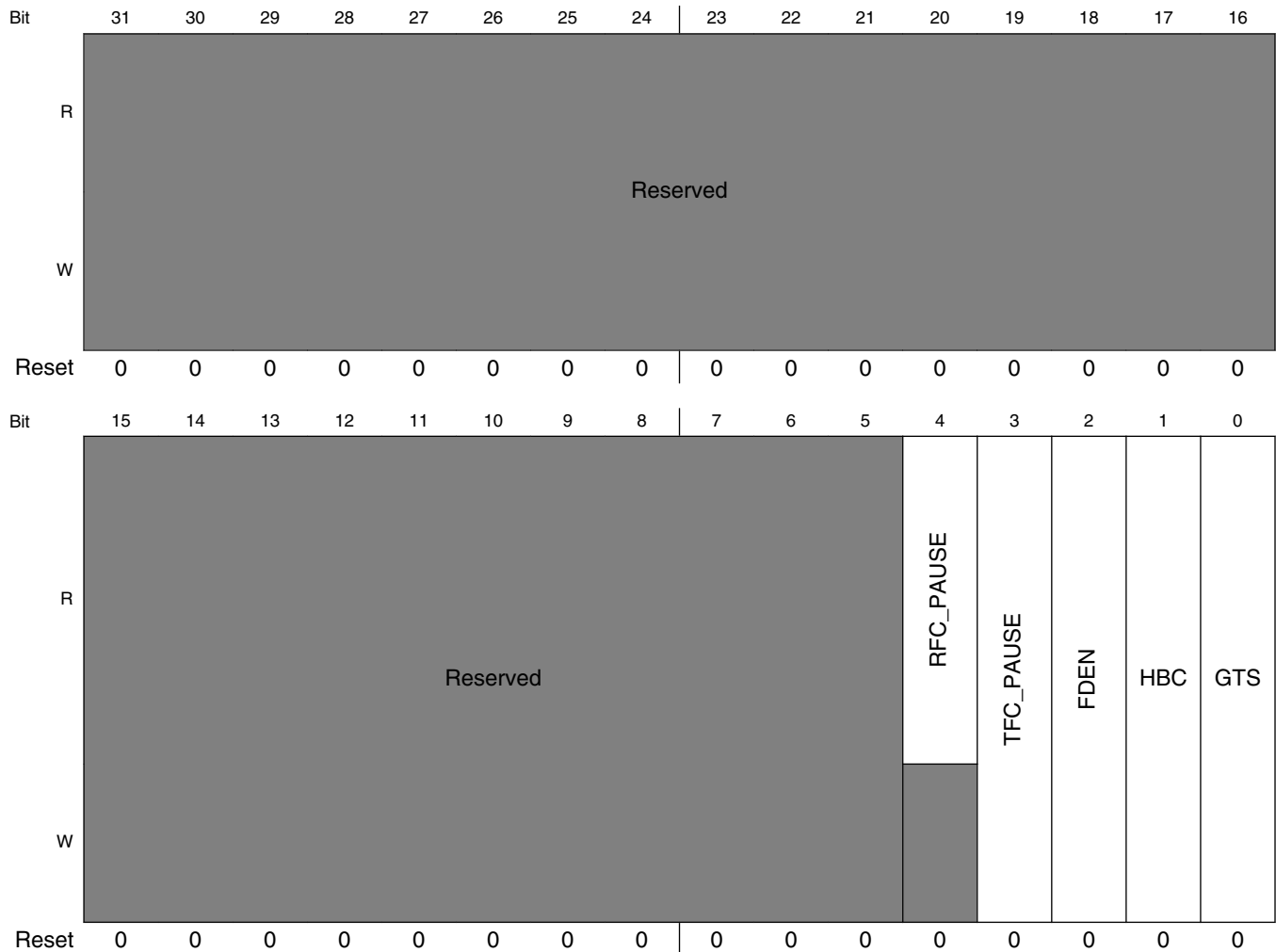
Field	Description
4 BC_REJ	Broadcast frame reject. When BC_REJ is set to 1, frames with DA (destination address) = 0xFF_FF_FF_FF_FF_FF are rejected unless the PROM bit is set to 1. If both BC_REJ and PROM are set to 1, then frames with broadcast DA is accepted and the M (MISS) bit is set in the receive buffer descriptor.
3 PROM	Promiscuous mode. All frames are accepted regardless of address matching.
2 MII_MODE	Media independent interface mode. Setting this bit to 1 selects MII mode, setting this bit equal to 1 selects 7-wire mode (used only for serial 10 Mbps). This bit controls the external interface mode for both transmit and receive blocks.
1 DRT	Disable receive on transmit. 0 Receive path operates independently of transmit (use for full-duplex or to monitor transmit activity in half-duplex mode). 1 Disable reception of frames while transmitting (normally used for half-duplex mode).
0 LOOP	Internal loopback. When LOOP is set to 1, transmitted frames are looped back internal to the device and the transmit output signals are not asserted. The system clock is substituted for the FEC_TX_CLK when LOOP is set to 1. DRT must be set to 0 when setting LOOP to 1.



## 24.8.10 Transmit control register (FEC\_TCR)

This register is read/write register which is written by the user to configure the transmit block. Bits [2:1] must only be modified when FEC\_ECR[ETHER\_EN] = 0 (that is, during initialization). This register is cleared at system reset.

Address: 218\_8000h base + C4h offset = 218\_80C4h



**FEC\_TCR field descriptions**

Field	Description
31–5 -	This field is reserved. Reserved read as 0
4 RFC_PAUSE	Receive frame control pause. This read-only status bit is set to 1 when a full-duplex flow control pause frame has been received and the transmitter is paused for the duration defined in this pause frame. This bit automatically clears when the pause duration is complete.

*Table continues on the next page...*

**FEC\_TCR field descriptions (continued)**

Field	Description
	0 Transmitter is not paused 1 Transmitter is paused after reception of full-duplex flow control pause frame
3 TFC_PAUSE	Transmit frame control pause. When this bit is set to 1, a pause frame is transmitted according to the following steps: <ol style="list-style-type: none"> <li>1. FEC stops transmission of data frames after the current transmission is complete.</li> <li>2. The GRA interrupt in the FEC_EIR register is asserted.</li> <li>3. With transmission of data frames stopped, the FEC transmits a MAC control pause frame.</li> <li>4. The FEC clears the TFC_PAUSE bit and resume transmitting data frames.</li> </ol> The FEC can still transmit a MAC control pause frame when the transmitter is paused due to user assertion of GTS or reception of a pause frame. 0 No pause frame is transmitted 1 Pause frame is transmitted
2 FDEN	Full duplex enable. When FDEN is set to 1, frames are transmitted independent of carrier sense and collision inputs. This bit must only be modified when ETHER_EN is cleared. 0 Full duplex is not enabled 1 Full duplex is enabled
1 HBC	Heartbeat control. When HBC is set to 1, the heartbeat check is performed after end of transmission and the HB bit in the status register is set if the collision input does not assert within the heartbeat window. This bit must only be modified when ETHER_EN is cleared. 0 Heartbeat check is not performed after end of transmission 1 Heartbeat check is performed after end of transmission
0 GTS	Graceful transmit stop. When GTS is set to 1, the FEC stops transmission after any frame that is currently being transmitted is completed and the GRA interrupt in the FEC_EIR register is asserted. If frame transmission is not currently underway, the GRA interrupt is asserted immediately. After transmission has completed, a "restart" can be accomplished by clearing the GTS bit. The next frame in the transmit FIFO is then transmitted. If an early collision occurs during transmission when GTS = 1, transmission stops after the collision. The frame is transmitted again after GTS is cleared. There can be old frames in the transmit FIFO that is transmitted when GTS is reasserted. To avoid this, clear FEC_ECR[ETHER_EN] following the GRA interrupt. 0 Graceful transmit stop is not enabled 1 Graceful transmit stop is enabled.

### 24.8.11 Physical address low register (FEC\_PALR)

The FEC\_PALR, which is written by the user, contains the lower 32 bits (bytes 0,1,2,3) of the 48-bit address used in the address recognition process to check for possible match between the DA field of receive frames and an individual DA. This register is also used for bytes 0 through 3 of the 6-byte source address field when transmitting pause frames. This register is unaffected by reset and must be initialized by the user.

Address: 218\_8000h base + E4h offset = 218\_80E4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PADDR1																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### FEC\_PALR field descriptions

Field	Description
31–0 PADDR1	Bytes 0 (bits 31:24), 1 (bits 23:16), 2 (bits 15:8) and 3 (bits 7:0) of the 6-byte individual address to be used for exact match, and the source address field in pause frames.

### 24.8.12 Physical address upper register (FEC\_PAUR)

The FEC\_PAUR, which is written by the user, and contains the upper 16 bits (bytes 4 and 5) of the 48-bit address used in the address recognition process to check for possible match between the DA field of receive frames and an individual DA. In addition, this register is used in bytes 4 and 5 of the 6-byte source address field when transmitting pause frames. Bits 15:0 of FEC\_PAUR are a constant type field (0x8808) which is used for transmission of pause frames. This register is unaffected by reset, and bits 31:16 must be initialized by the user.

Address: 218\_8000h base + E8h offset = 218\_80E8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PADDR2																TYPE															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0

**FEC\_PAUR field descriptions**

Field	Description
31–16 PADDR2	Bytes 4 (bits 31:24) and 5 (bits 23:16) of the 6-byte individual address to be used for exact match, and the source address field in pause frames.
15–0 TYPE	Type field in pause frames. This field has a constant value of 0x8808.

**24.8.13 Opcode and pause duration register (FEC\_OPDR)**

FEC\_OPDR contains the 16-bit Opcode, and 16-bit pause duration fields used in transmission of a pause frame. The Opcode field is a constant value, 0x0001. When another node detects a pause frame, that node pauses transmission for the duration specified in the pause duration field. This register is not reset and must be initialized by the user.

Address: 218\_8000h base + ECh offset = 218\_80ECh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OPCODE																PAUSE_DUR															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**FEC\_OPDR field descriptions**

Field	Description
31–16 OPCODE	Opcode field used in pause frames. These bits are a constant, 0x0001.
15–0 PAUSE_DUR	Pause duration field used in pause frames.

### 24.8.14 Descriptor individual address upper register (FEC\_IAUR)

The FEC\_IAUR, which is written by the user, contains the upper 32 bits of the 64-bit individual address hash table used in the address recognition process to check for possible match between the DA field of receive frames and an individual DA. This register is unaffected by reset, and must be initialized by the user.

Address: 218\_8000h base + 118h offset = 218\_8118h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IADDR1																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### FEC\_IAUR field descriptions

Field	Description
31–0 IADDR1	The upper 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a unicast address. Bit 31 of IADDR1 contains hash index bit 63. Bit 0 of IADDR1 contains hash index bit 32.

### 24.8.15 Descriptor individual address lower register (FEC\_IALR)

The FEC\_IALR, which is written by the user, contains the lower 32 bits of the 64-bit individual address hash table used in the address recognition process to check for possible match with the DA field of receive frames with an individual DA. This register is unaffected by reset, and must be initialized by the user.

Address: 218\_8000h base + 11Ch offset = 218\_811Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IADDR2																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### FEC\_IALR field descriptions

Field	Description
31–0 IADDR2	The lower 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a unicast address. Bit 31 of IADDR2 contains hash index bit 31. Bit 0 of IADDR2 contains hash index bit 0.

## 24.8.16 Descriptor group address upper register (FEC\_GAUR)

The FEC\_GAUR, which is written by the user, contains the upper 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a multicast address. This register must be initialized by the user.

Address: 218\_8000h base + 120h offset = 218\_8120h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GADDR1																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### FEC\_GAUR field descriptions

Field	Description
31–0 GADDR1	The GADDR1 register contains the upper 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a multicast address. Bit 31 of GADDR1 contains hash index bit 63. Bit 0 of GADDR1 contains hash index bit 32.

## 24.8.17 Descriptor group address lower register (FEC\_GALR)

The FEC\_GALR, which is written by the user, contains the lower 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a multicast address. This register must be initialized by the user.

Address: 218\_8000h base + 124h offset = 218\_8124h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GADDR2																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### FEC\_GALR field descriptions

Field	Description
31–0 GADDR2	The GADDR2 register contains the lower 32 bits of the 64-bit hash table used in the address recognition process for receive frames with a multicast address. Bit 31 of GADDR2 contains hash index bit 31. Bit 0 of GADDR2 contains hash index bit 0.

## 24.8.18 Transmit FIFO watermark register (FEC\_TFWR)

The FEC\_TFWR is programmed by the user to control the amount of data required in the transmit FIFO before transmission of a frame can begin. This allows the user to minimize transmit latency (FEC\_TFWR[1:0] = 0x *n* ) or allow for larger bus access latency (FEC\_TFWR[1:0] = 11) due to contention for the system bus. Setting the watermark to a high value minimizes the risk of transmit FIFO underrun due to contention for the system bus. In some use cases the byte counts associated with the FEC\_TFWR field need to be modified to match system requirements, such as the worst-case bus access latency by the transmit data DMA channel.

Address: 218\_8000h base + 144h offset = 218\_8144h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															X_WMRK
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

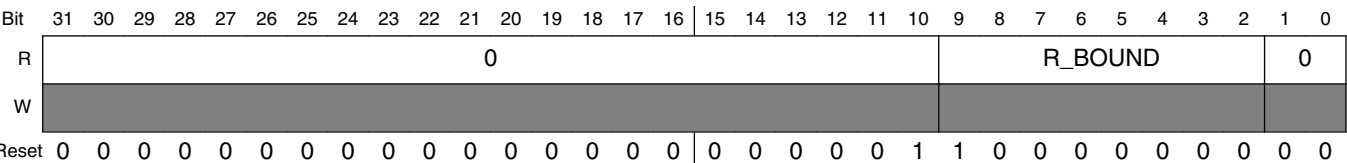
### FEC\_TFWR field descriptions

Field	Description
31–2 -	This field is reserved. Reserved, read as 0
1–0 X_WMRK	Number of bytes written to transmit FIFO before transmission of a frame begins  0x 64 bytes written 10 128 bytes written 11 192 bytes written

24.8.19 FIFO receive bound register (FEC\_FRBR)

The FEC\_FRBR register can be read to determine the upper address bound of the FIFO RAM. Drivers can use this value, along with the FEC\_FRSR to appropriately divide the available FIFO RAM between the transmit and receive data paths.

Address: 218\_8000h base + 14Ch offset = 218\_814Ch



FEC\_FRBR field descriptions

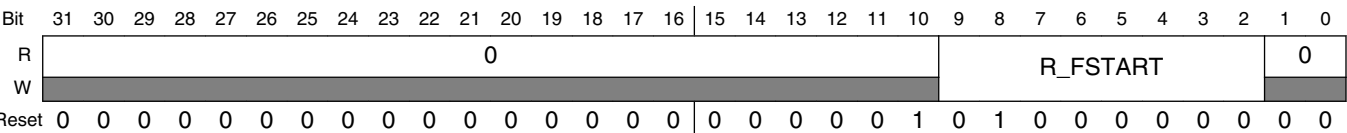
Field	Description
31–10 Reserved	This read-only field is reserved and always has the value 0.
9–2 R_BOUND	Read-only. Highest valid FIFO RAM address.
1–0 Reserved	This read-only field is reserved and always has the value 0.

24.8.20 FIFO receive FIFO start registers (FEC\_FRSR)

FEC\_FRSR is an 8-bit register programmed by the user to indicate the starting address of the receive FIFO. FEC\_FRSR marks the boundary between the transmit and receive FIFOs. The transmit FIFO uses addresses from the start of the FIFO to the location four bytes before the address programmed into the FEC\_FRSR. The receive FIFO uses the addresses from FEC\_FRSR to FEC\_FRBR inclusive.

The default value of the receive FIFO starting address is 0x40. This is the value assigned by hardware at reset.

Address: 218\_8000h base + 150h offset = 218\_8150h





**FEC\_FRSR field descriptions**

Field	Description
31–10 Reserved	This read-only field is reserved and always has the value 0.
9–2 R_FSTART	Address of first receive FIFO location. Acts as delimiter between receive and transmit FIFOs.
1–0 Reserved	This read-only field is reserved and always has the value 0.

### 24.8.21 Receive buffer descriptor ring start register (FEC\_ERDSR)

The register, which is written by the user, provides a pointer to the start of the circular receive buffer descriptor queue in external memory. This pointer must be 128-bit aligned (that is, evenly divisible by 16).

This register is unaffected by reset and must be initialized by the user prior to operation.

Address: 218\_8000h base + 180h offset = 218\_8180h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	R_DES_START															
W																
Reset	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	R_DES_START														Reserved	
W																
Reset	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*	u*

\* Notes:

- u = Unaffected by reset.

**FEC\_ERDSR field descriptions**

Field	Description
31–2 R_DES_START	Pointer to start of receive buffer descriptor queue.
1–0 -	This field is reserved. Reserved, read as 0

## 24.8.22 Transmit buffer descriptor ring start register (FEC\_ETDSR)

The register, which is written by the user, provides a pointer to the start of the circular transmit buffer descriptor queue in external memory. This pointer must be 128-bit aligned (that is, evenly divisible by 16).

This register is unaffected by reset and must be initialized by the user prior to operation.

Address: 218\_8000h base + 184h offset = 218\_8184h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	X_DES_START															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	X_DES_START														Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### FEC\_ETDSR field descriptions

Field	Description
31–2 X_DES_START	Pointer to start of transmit buffer descriptor queue.
1–0 -	This field is reserved. Reserved, read as 0

## 24.8.23 Maximum receive buffer size register (FEC\_EMRR)

The FEC\_EMRR is a user-programmable register which dictates the maximum size of all receive buffers. Note that receive frames is truncated at 2k-1(2047) bytes, so bits 31-11 are not used. The programmed value accounts for the fact that the receive CRC is always written into the last receive buffer. To allow one maximum size frame per buffer, FEC\_EMRR must be set to FEC\_RCR[MAX\_FL] or larger. The FEC\_EMRR must be evenly divisible by 16. To ensure this, bits 3-0 are forced to low, and hence only bits 10-4 are actually used. To minimize bus utilization (descriptor fetches), it is recommended that FEC\_EMRR must be greater than or equal to 256 bytes.

The FEC\_EMRR is unaffected by reset, and must be initialized by the user.

Address: 218\_8000h base + 188h offset = 218\_8188h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																R_BUF_SIZE						Reserved									
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**FEC\_EMRR field descriptions**

Field	Description
31–11 -	This field is reserved. Reserved, is written to 0 by the host processor.
10–4 R_BUF_SIZE	Receive buffer size.
3–0 -	This field is reserved. Reserved, is written to 0 by the host processor.



# Chapter 25

## General Power Controller (GPC)

### 25.1 Overview

The General Power Control (GPC) block includes the sub-blocks listed here.

- [DVFS - Dynamic Voltage & Frequency Scaling](#) load tracking for CPU
- [CPU Power Gating Control \(PGC\)](#)
- GPU and Display power gating controller

Each sub-block has its own IP registers.

GPC determines wake-up irq for exiting STOP mode (with or without CPU power gating).

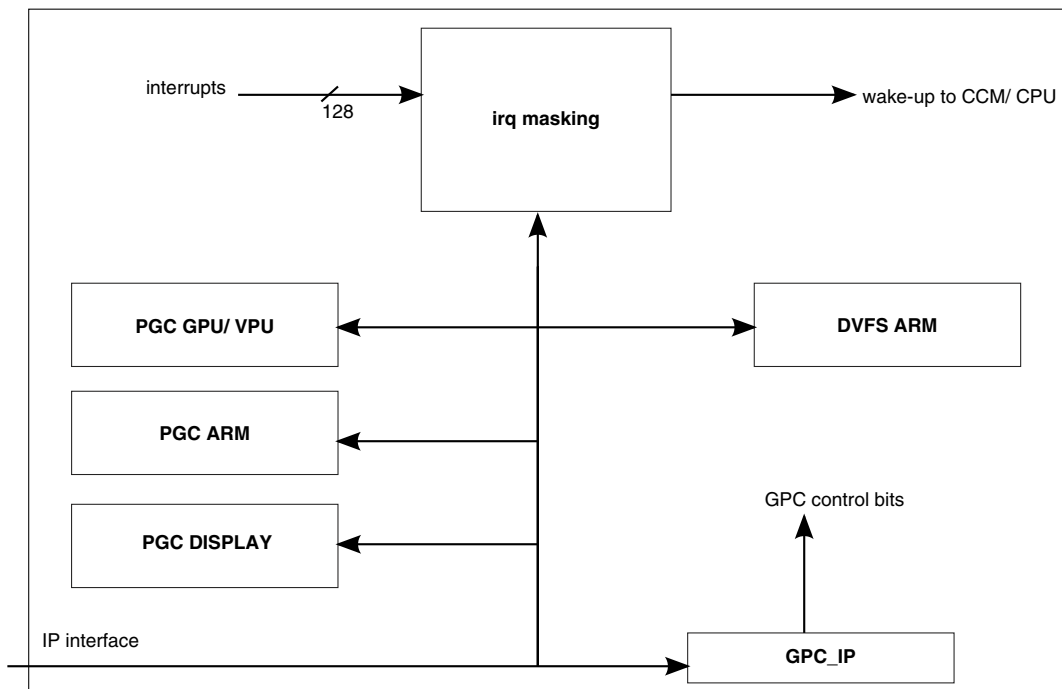


Figure 25-1. GPC Block Diagram

## 25.2 Clocks

The table found here describes the clock sources for GPC.

Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 25-1. GPC Clocks**

Clock name	Clock Root	Description
ipg_clk	ipg_clk_root	Peripheral clock
ipg_clk_s	ipg_clk_root	Peripheral access clock
pgc_clk	ipg_clk_root	PGC peripheral clock
sys_clk	ipg_clk_root	Module clock

## 25.3 DVFS overview

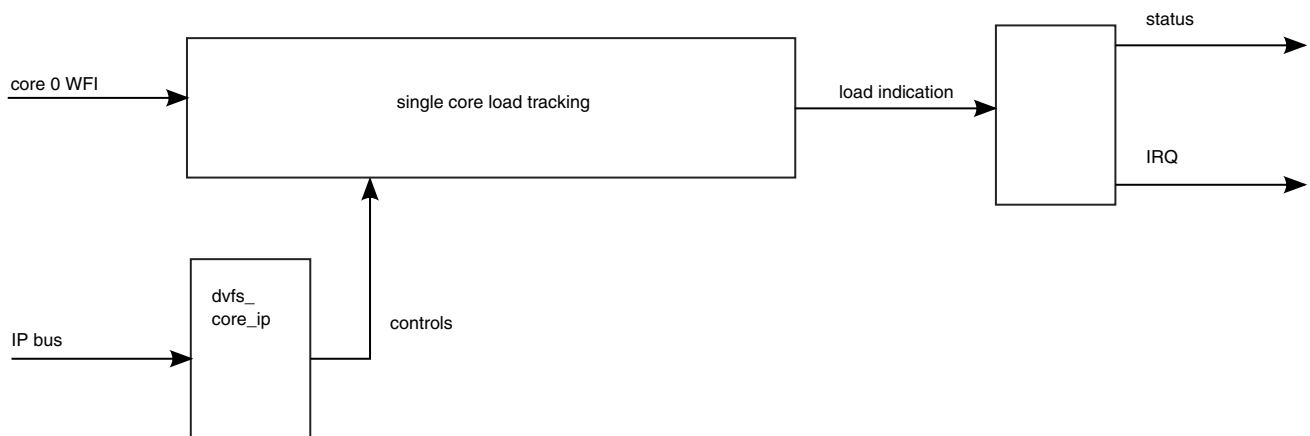
The DVFS allows simple dynamic voltage frequency scaling.

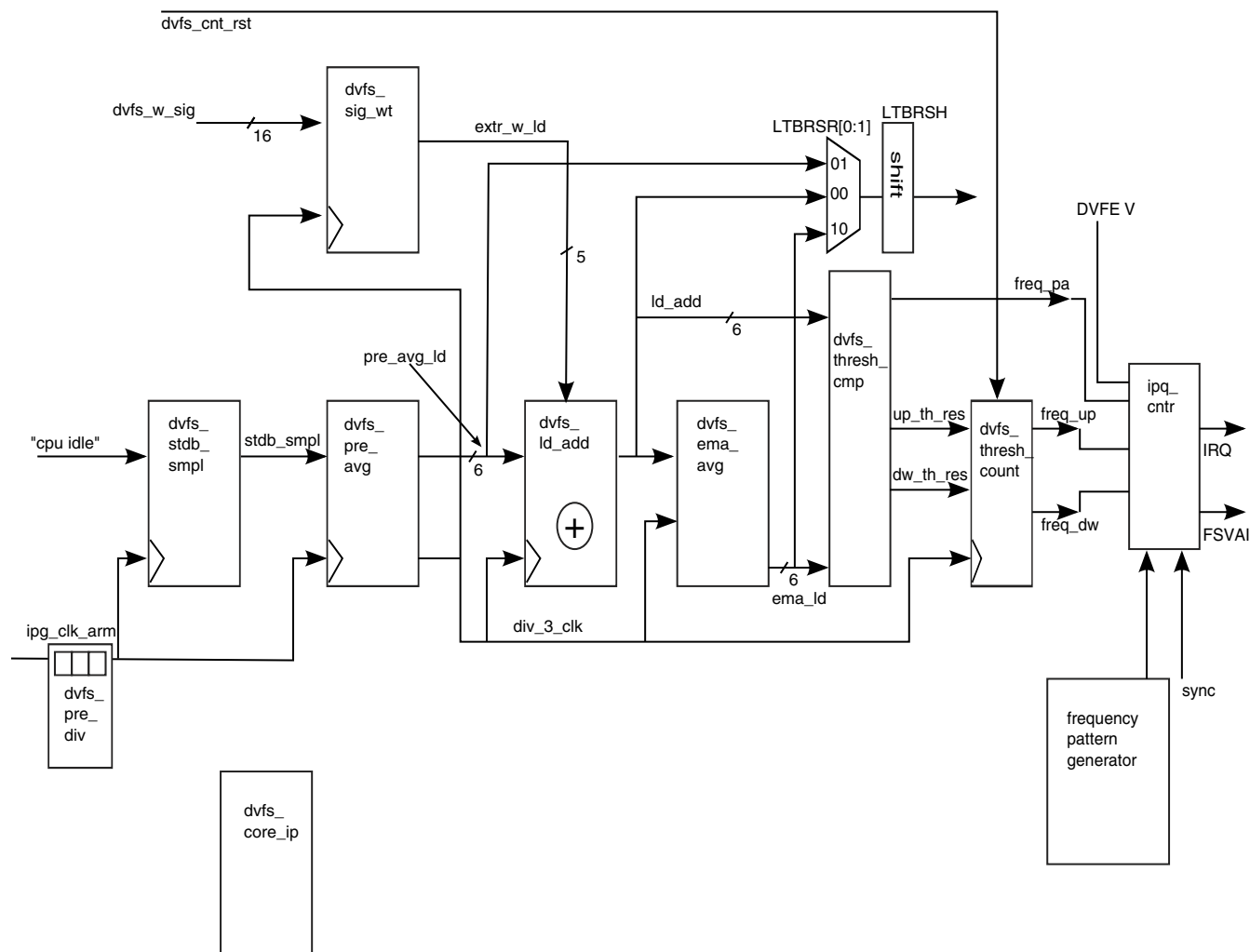
The frequency of the core clock domain and the voltage of the core power domain can be changed on the fly while all modules (including the MCU) continue their normal operation. The frequency of the core clock domain can be changed by temporally switching to an alternate PLL clock, or by changing the post dividers division factors.

The DVFS load tracking block allows hardware tracking on the core load and generate an interrupt when a frequency change is requested.

### NOTE

DVFS is a monitor that only provides an interrupt when CPU load exceeds the predefined value and does not send any request to make a change of voltage and frequency. This can be done by the user in the CPU interrupt routine or SDMA routine.

**Figure 25-2. DVFS\_core diagram**



**Figure 25-3. DVFS\_core: single core load tracking module**

### 25.3.1 Features

The DVFS load tracking block includes the following features:

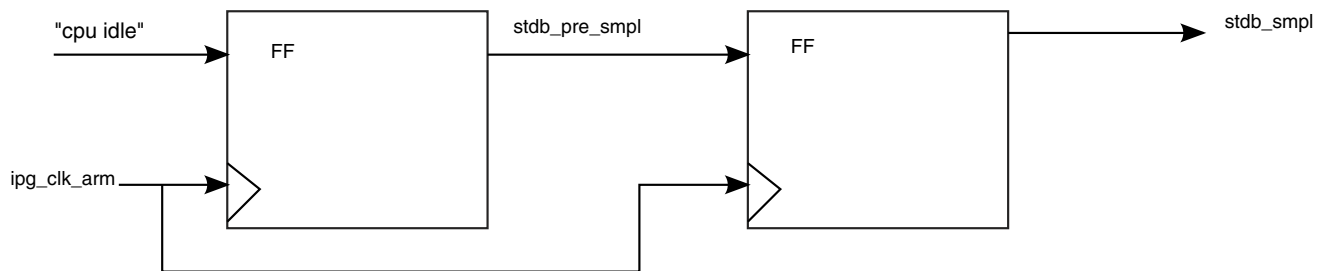
- Configurable include/exclude of input signals:
  - per-core ARM standby signal (idle / non-idle)
  - 16 general purpose bits (common for all 4 tracking modules)
    - Configurable weight to each bit.
- Configurable generated clocks and averaging time slicing (respond time).
- Configurable panic mode respond logic (for frequency up).
- Programmable buffer for last 4, 8, 12, or 16 load tracking samples of core 0 (only).



## 25.4 Component Blocks description

### 25.4.1 dvfs\_stdb\_smpl block

This block samples the `ccm_arm_stdb` signal (ARM STANDBYWFI signal - idle state indicating) according to the edge of the `ipg_clk_arm` (ARM system clock) signal.



**Figure 25-4. dvfs\_stdb\_smpl block diagram**

This block synchronizes the `ccm_arm_stdb` signal with the `ipg_clk_arm` clock. The two signals enter the flip-flops twice and become synchronized. The resulting synchronized signal is `stdb_smpl`.

## 25.4.2 dvfs\_sig\_wt block

The purpose of this block is to sample the 16 GeP (general purpose) load signals, multiply each one of them by appropriate weight and sum products.

The sampling is done by the slow clock div\_3\_clk.

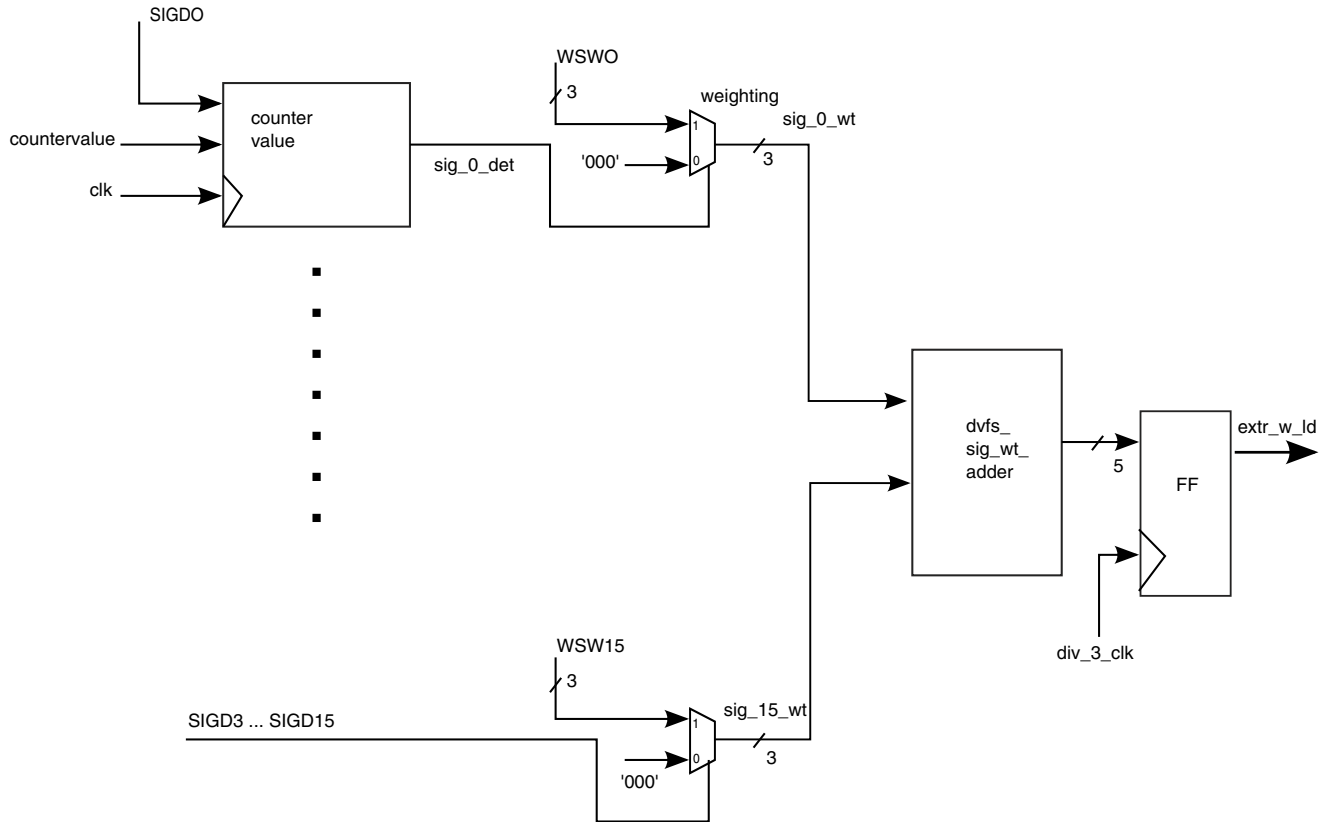


Figure 25-5. dvfs\_sig\_wt block diagram

## 25.4.3 dvfs\_pre\_avg block

The purpose of the dvfs\_pre\_avg block is to perform simple, non-overlapping averaging, reducing the sampling clock frequency and providing a level-based average index of the tracked CPU load.

External signals:

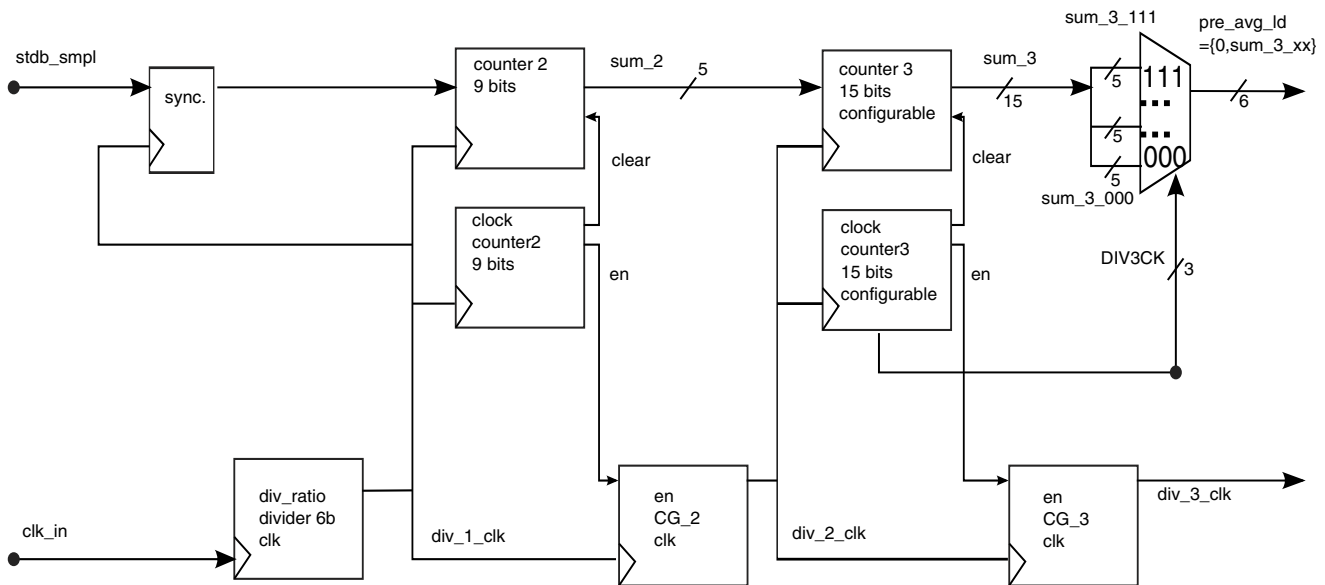
- stdb\_sampl - (input) sampled ARM standby signal
- ipg\_clk\_arm - (input) ARM system clk
- pre\_avg\_ld[4..0] - (output) averaged load
- div\_3\_clk - (output) clock, generated (reduced) from div\_2\_clk

- div\_2\_clk - (output) clock, generated (reduced) from div\_1\_clk
- div\_1\_clk - (output) clock, generated (reduced) from ipg\_clk\_arm

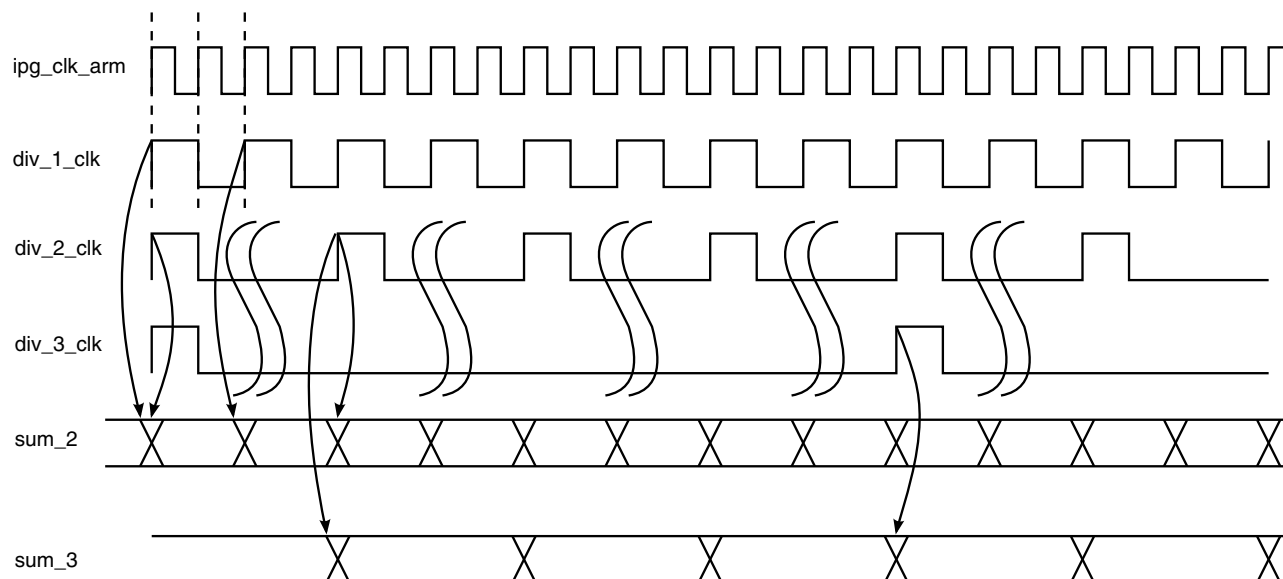
In the current implementation, the averaging is performed in three stages:

1. The first stage !IDLE is sampled by divided sys\_clock.
2. The second stage (counter 2) performs an averaging operation with constant parameters. The counter is a 9-bit adder whose output is sampled by the div\_2\_clk clock. The five highest bits of the sum\_2 signal are passed to counter 3 (equal to shift right operation).
3. The third (last) stage (counter 3) performs an averaging operation with configurable parameters, set by the DIV3CK. The counter 3 cell is a 15-bit adder with output sampled by div\_3\_clk. The Pre\_avg\_shifter\_5 cell passes configurable bits from the sum\_3 signal.

The output 5 bits avg\_load signal provides a result with a tolerance of ~3%. The supported values range is 0-0.97.



**Figure 25-6. dvfs\_pre\_avg block diagram**

**dvfs\_pre\_avg clocks****Figure 25-7. dvfs\_pre\_avg clocks**

The internal clocks in **dvfs\_avg\_pre** block are generated from **ipg\_clk\_arm**. The details are in the tables below. For **div\_3\_clk** averaging, see [DVFS Control \(DVFSC\\_CNTR\)](#) for more information.

**Table 25-2. dvfs\_pre\_avg signals and its values**

signal	binary max value	binary min value	decimal max value	decimal min value
sum_2	11111	0000	31	0
sum_3	111'1100'0000'0000	000'0000'0000'0000	31744	0
pre_avg_load	11111	0000	31	0

**Table 25-3. dvfs\_pre\_avg generated clocks**

clock name	generated from	ratio to source	ratio to ipg_clk_arm	max clk freq.	note
ipg_clk_arm	N/A	N/A	1	66MHz	source clock
div_1_clk	ipg_clk_arm	configurable	configurable	66MHz	configurable
div_2_clk (gated)	div_1_clk	512	configurable	128KHz	none
div_3_clk (gated)	div_2_clk	configurable	configurable	128KHz	configurable

### 25.4.4 dvfs\_ld\_add block

The dvfs\_ld\_add block sums the CPU load tracked by idle/non-idle signal slot, and the load detected from the additional load signals weighted by dvfs\_sig\_wt block.

The adder should perform the following operation:

$$\text{extr\_w\_ld}[4..0] + \{0, \text{pre\_avg\_ld}[4..0]\}$$

The input signals of five bits produce output signal of six bits, providing 3% resolution in the range of 0-1.97 of load indication. (1 is equal to 100% load tracking with no additional signals include.)

The output ld\_add[5..0] is sampled by div\_3\_clk signal.

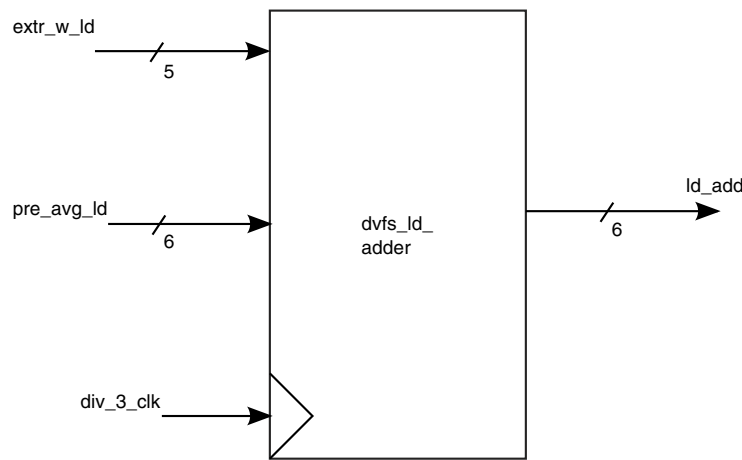


Figure 25-8. dvfs\_ld\_add block diagram

### 25.4.5 dvfs\_ema\_avg block

The purpose of dvfs\_ema\_avg (EMA - Exponential Moving Average) block is to calculate an exponential moving average of the tracked CPU load.

The parameters of EMA are defined by EMAC bits in DVFSEMAC. These parameters set how many samples of simple average will be taken into account.

The EMA formula is:  $\text{EMA}(i) = a \cdot X(i) + (1-a) \cdot \text{EMA}(i-1)$

where:

$$a = 2 / (N+1);$$

N is the number of samples taken into account.

X(i) = current input sample;

$EMA(i-1)$  = previous value of EMA.

By setting the value of "a", the behavior of EMA is defined.

In the following table, the parameter "a" is listed relatively to the amount of the X(i) samples taken into account ("N") in the equation above: (the resolution of the digital multiplier is limited, hence the lowest values of the "a" can be linked to a range of included samples in EMA instead of single number).

The following table also contains the amount of cycles in which the lower frequency will be masked from the moment that DVFS is enabled (and not between frequency switches), so that enough information about the history can be gained before the frequency is lowered.

**Table 25-4. EMA settings**

samples included in EMA calculatio n (N)	Number of cycles while lower frequency interrupt is masked	"a" value (decimal)	"a" value, adjusted by binary resolution	"a" value (binary)								
				bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7	bit 8
-----		-----	-----									
1	6	1.000	1.000	1	0	0	0	0	0	0	0	0
2	11	0.667	0.668	0	1	0	1	0	1	0	1	1
3	11	0.500	0.500	0	1	0	0	0	0	0	0	0
4	22	0.400	0.398	0	0	1	1	0	0	1	1	0
5	22	0.333	0.332	0	0	1	0	1	0	1	0	1
6	22	0.286	0.285	0	0	1	0	0	1	0	0	1
7	22	0.250	0.250	0	0	1	0	0	0	0	0	0
8	43	0.222	0.223	0	0	0	1	1	1	0	0	1
9	43	0.200	0.199	0	0	0	1	1	0	0	1	1
10	43	0.182	0.180	0	0	0	1	0	1	1	1	0
11	43	0.167	0.168	0	0	0	1	0	1	0	1	1
12	43	0.154	0.152	0	0	0	1	0	0	1	1	1
13	43	0.143	0.141	0	0	0	1	0	0	1	0	1
14	43	0.133	0.133	0	0	0	1	0	0	0	1	0
15	43	0.125	0.125	0	0	0	1	0	0	0	0	0
16	86	0.118	0.117	0	0	0	0	1	1	1	1	0
17	86	0.111	0.109	0	0	0	0	1	1	1	0	0
18	86	0.105	0.105	0	0	0	0	1	1	0	1	1
19	86	0.100	0.102	0	0	0	0	1	1	0	1	0
20	86	0.095	0.094	0	0	0	0	1	1	0	0	0
21	86	0.091	0.090	0	0	0	0	1	0	1	1	1
22	86	0.087	0.086	0	0	0	0	1	0	1	1	0

*Table continues on the next page...*

Table 25-4. EMA settings (continued)

samples included in EMA calculatio n (N)	Number of cycles while lower frequency interrupt is masked	"a" value (decimal)	"a" value, adjusted by binary resolution	"a" value (binary)								
23	86	0.083	0.082	0	0	0	0	1	0	1	0	1
25	86	0.077	0.078	0	0	0	0	1	0	1	0	0
26	86	0.074	0.074	0	0	0	0	1	0	0	1	1
28	86	0.069	0.070	0	0	0	0	1	0	0	1	0
30	86	0.065	0.066	0	0	0	0	1	0	0	0	1
31	86	0.063	0.063	0	0	0	0	1	0	0	0	0
33	173	0.059	0.059	0	0	0	0	0	1	1	1	1
35	173	0.056	0.055	0	0	0	0	0	1	1	1	0
38	173	0.051	0.051	0	0	0	0	0	1	1	0	1
42	173	0.047	0.047	0	0	0	0	0	1	1	0	0
45	173	0.043	0.043	0	0	0	0	0	1	0	1	1
50	173	0.039	0.039	0	0	0	0	0	1	0	1	0
56	173	0.035	0.035	0	0	0	0	0	1	0	0	1
~64	173	0.031	0.031	0	0	0	0	0	1	0	0	0
~74	256	0.027	0.027	0	0	0	0	0	0	1	1	1
~86	256	0.023	0.023	0	0	0	0	0	0	1	1	0
~100	256	0.020	0.020	0	0	0	0	0	0	1	0	1
~128	256	0.016	0.016	0	0	0	0	0	0	1	0	0
~170	512	0.012	0.012	0	0	0	0	0	0	0	1	1
~260	512	0.008	0.008	0	0	0	0	0	0	0	1	0
~500	512	0.004	0.004	0	0	0	0	0	0	0	0	1
N/A		0.000	0.000	0	0	0	0	0	0	0	0	0

### dvfs\_ema\_avg block diagram

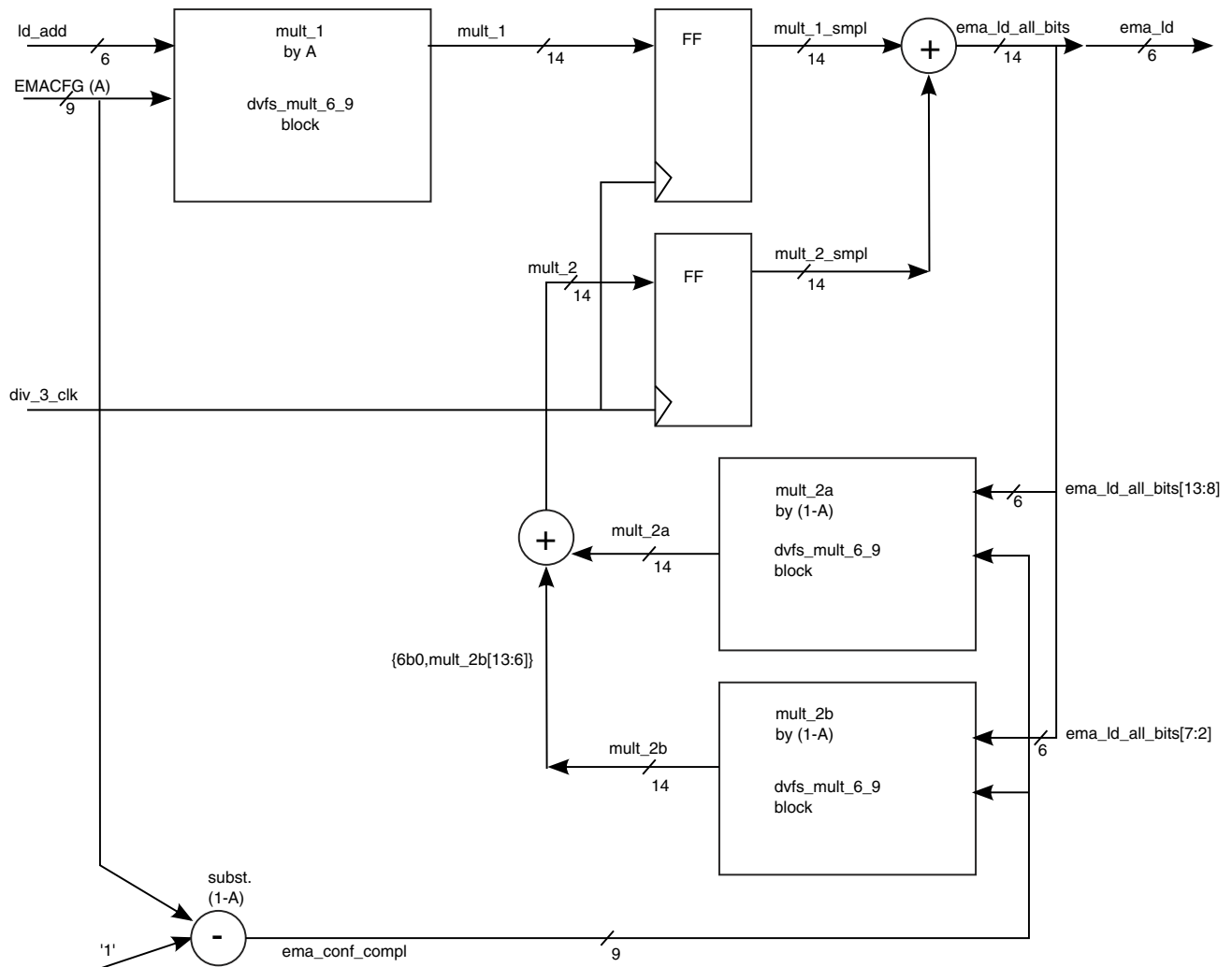


Figure 25-9. dvfs\_ema\_avg block diagram

The EMA block inputs are as follows:

- `ld_add[5..0]` - load level
- `EMACFG[8..0]` - "a" parameter of EMA algorithm
- `div_2_clk` - fast clock, not shown this in the diagram
- `div_3_clk` - slow clock

The EMA block outputs the following:

- `ema_ld[5..0]` - result of EMA algorithm (by `div_3_clk`)



The mult\_a block multiplies between comp\_load (6 bits) and EMACFG (9 bits). For the multiply operation, a faster clock (for internal sum) is required, which is provided by div\_2\_clk. Div\_2\_clk is faster than the div\_3\_clk by a factor of at least 16, which is enough for 6\*9 or 9\*6 operations. Only the highest six bits are taken from the result of mult\_a.

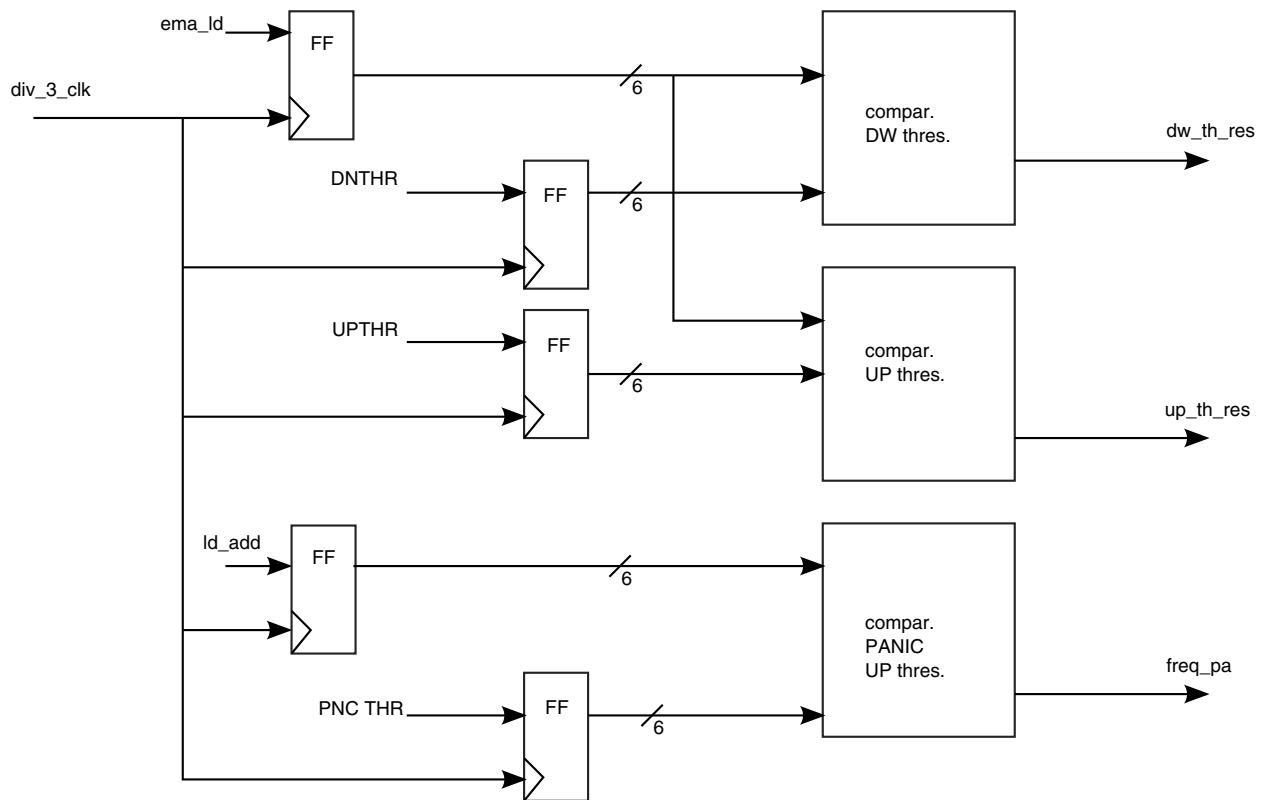
The mult\_b block operates in a similar manner.

The adder block's output is sampled with div\_3\_clk clock.

### 25.4.6 dvfs\_thres\_cmp block

The dvfs\_thres\_cmp block compares the CPU load value to programmable threshold levels.

The comparators as shown in the drawing below are used:



**Figure 25-10. dvfs\_tresh\_cmp diagram**

The block is composed of three (3) comparators:

1. compar\_dw\_thres comparator, which compares between ema\_ld signal (output of EMA block) and DNTHR signal (taken from config register, bits DNTHR).

2. compar\_up\_thres comparator, which compares between ema\_ld signal (output of EMA block) and UPTHR signal (taken from config register, bits UPTHR).
3. compar\_panic\_up\_thres comparator, which compares between ld\_add signal (output of load\_adder block) and PNCTHR signal (taken from config register, bits PNCTHR).

### **NOTE**

The current implementation of this block enables step-by-step down frequency change. For more advanced options, the number of the DW threshold comparators should be increased (up to three for four-level DVFS).

## **25.4.7 dvfs\_thresh\_count block**

The purpose of the dvfs\_thres\_count block is to count consecutive threshold overcomes of dw\_th\_res and up\_th\_res (outputs of threshold\_comp block).

If any of the counters (see the figure below) receives a null (zero) input synchronous with the clk3 signal, the counter is reset.

If the counter reaches a user defined value, its output is set to an active level. These counters are reset each time frequency scaling occurs or if the threshold overcomes are consecutive.

This block's output signals are saved in configuration register 0, bits [3,2].

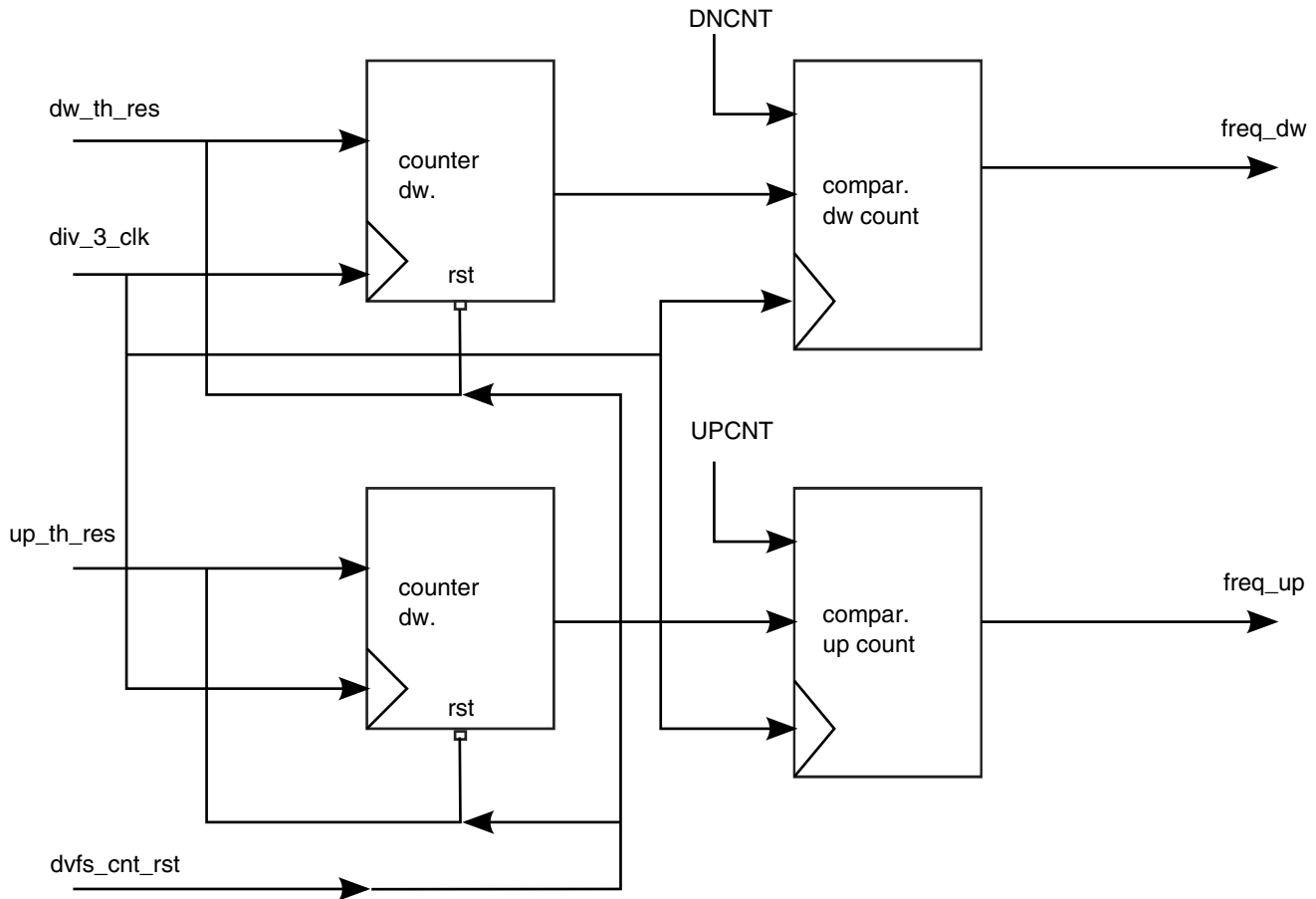


Figure 25-11. Thresh\_counters block diagram

### 25.4.8 Load Tracking Buffer Register

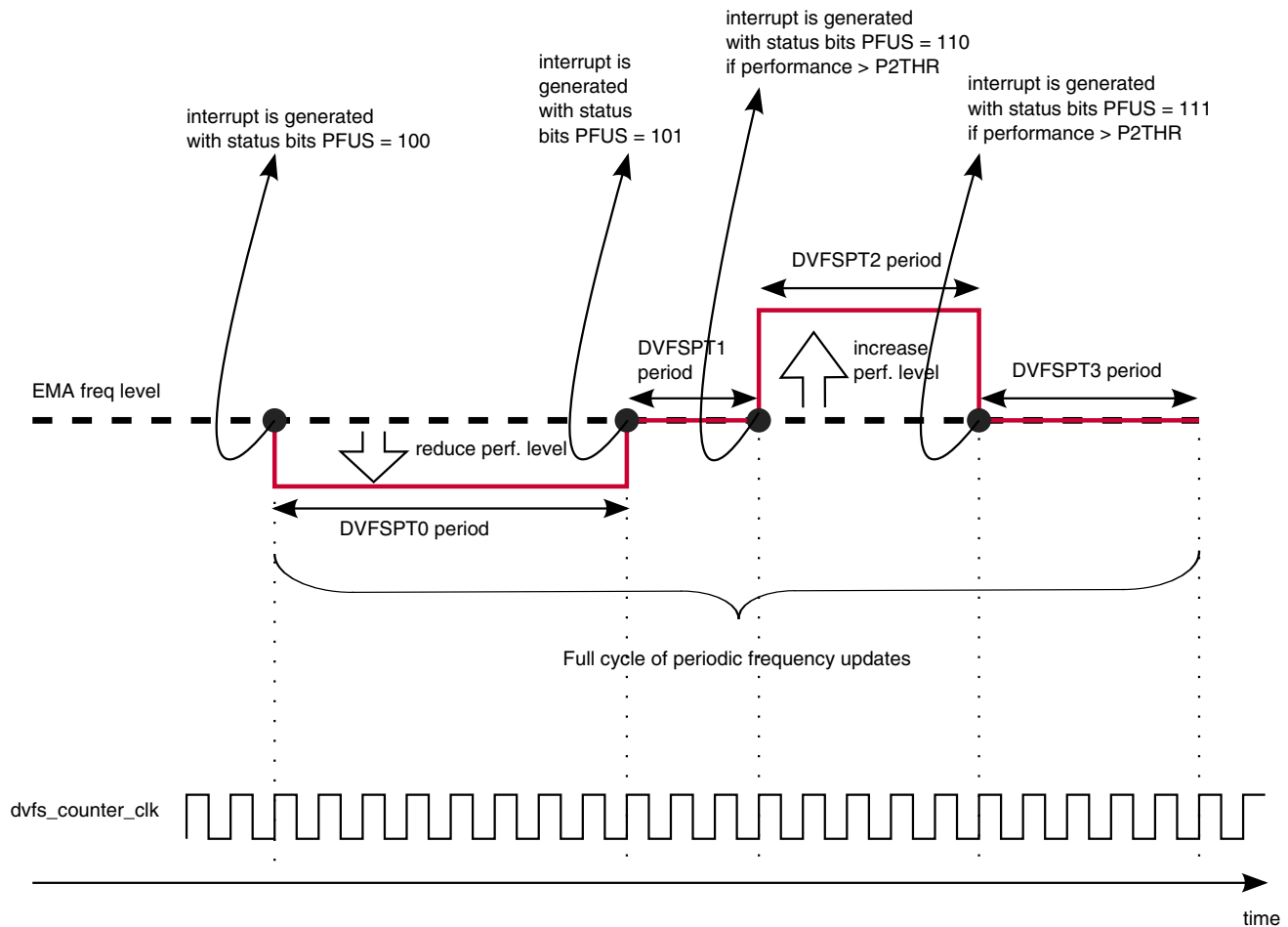
The purpose of the load tracking buffer register is to save last 16 samples of the tracked load (before EMA operation).

Hence, the 4 bits of `ld_add` signal (depending on the `LTBRSH`) are saved continuously, overwriting each time the latest sample. Each save is carried upon detecting an edge of the `div_3_clk` signal.

### 25.4.9 Frequency Pattern Generator

Frequency pattern generator is able to manage the frequency update requests periodically, additionally to main frequency update requests (if bit `FPUE` is set).

The periodic requests are created following the `DVFSPT0`, `DVFSPT1`, `DVFSPT2` and `DVFSPT3` register values, as described in the figure below.



**Figure 25-12. DVFS periodic frequency update requests**

In case that one of the DVFSPT0-DVFSPT3 period is set to "0", such frequency update will be skipped.

The periodic frequency update status is reflected in PFUS bits (reduce/increase performance request is an example - the actual steps taken upon period expiration are defined by s/w routine).

Dvfs\_counter\_clk is ckih divided 64:  $26\text{MHz}/64=0.40625\text{MHz}$ . The DVFSPT0, DVFSPT1, DVFSPT2 and DVFSPT3 counter are selected for 17 bits each to provide delay of  $2^{18}-1=262143$  counts, that are equal to 645ms. On the other hand, clk cycle of 0.40625MHz is  $\sim 2.46\mu\text{s}$ , that is fast enough to provide high resolution for frequency management for tasks.

The DVFSPT2 and DVFSPT3 interrupt generation is conditional upon current performance being greater than DVFSPT2[P2THR]. Otherwise the pattern delay will be counted, but without interrupt generation.

## 25.5 DVFS output event/interrupt configuration

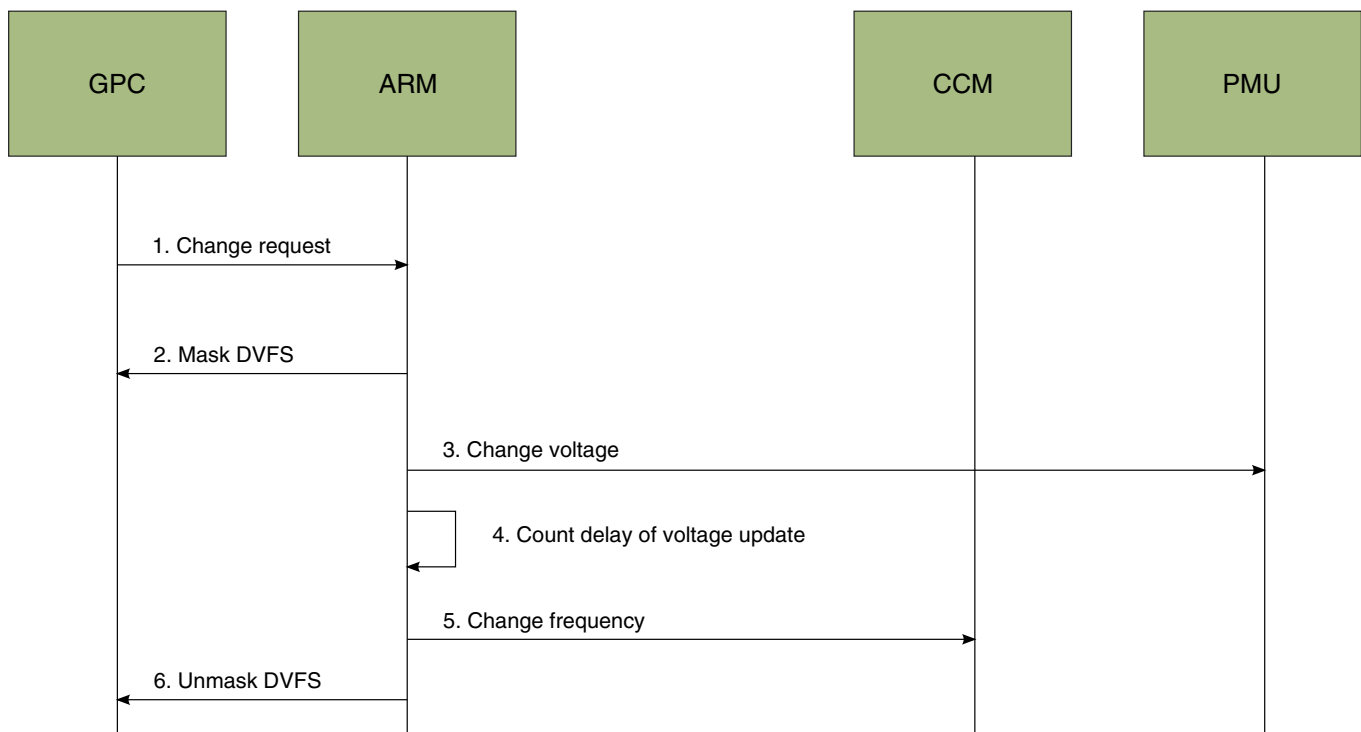
Event/interrupt will be always high as long as LBFL is '1' and is not cleared by SW. Unless DVFEV (always event) is asserted. Then the event/interrupt will be toggled up and down at every toggle of div\_3\_clk.

### 25.5.1 Interrupts

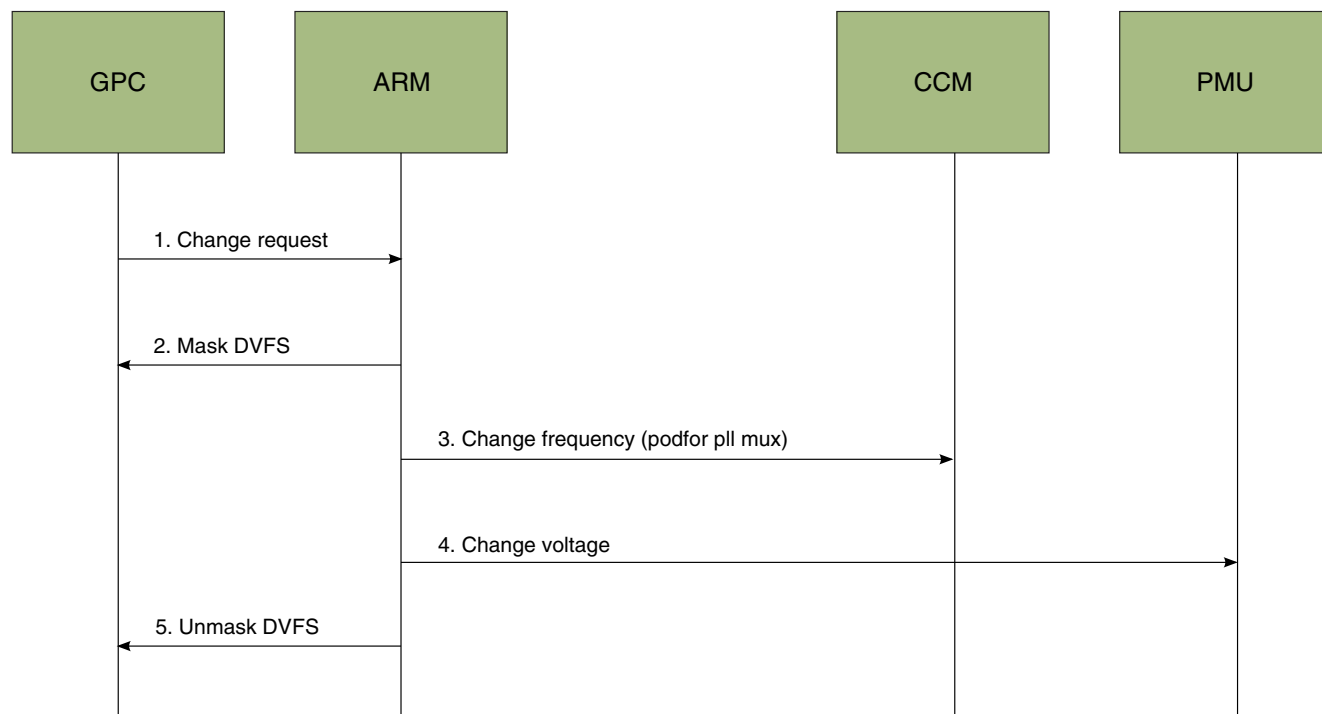
DVFS generates an interrupt that indicates that frequency and voltage update is needed. The user has to read the FSVAIM bits in order to know which change needs to be done.

### 25.5.2 DVFS Change Request Sequence Diagrams

The following figures describe the sequence on DVFS interrupt.



**Figure 25-13. DVFS - frequency increase**



**Figure 25-14. DVFS - frequency decrease**

CPU DVFS frequency change can be performed in 2 ways:

- PLL inputs muxing update
- clock dividers update

## 25.6 Power Gating Control (PGC)

Power Gating (PGC) is applied to the ARM CPU only in STOP low power mode, after all essential CPU registers data are saved by ARM dormant procedure.

If any of the unmasked interrupts appears, CPU is powered up and clock restore request (exit from STOP mode) is sent to CCM.

### PGC power down sequence:

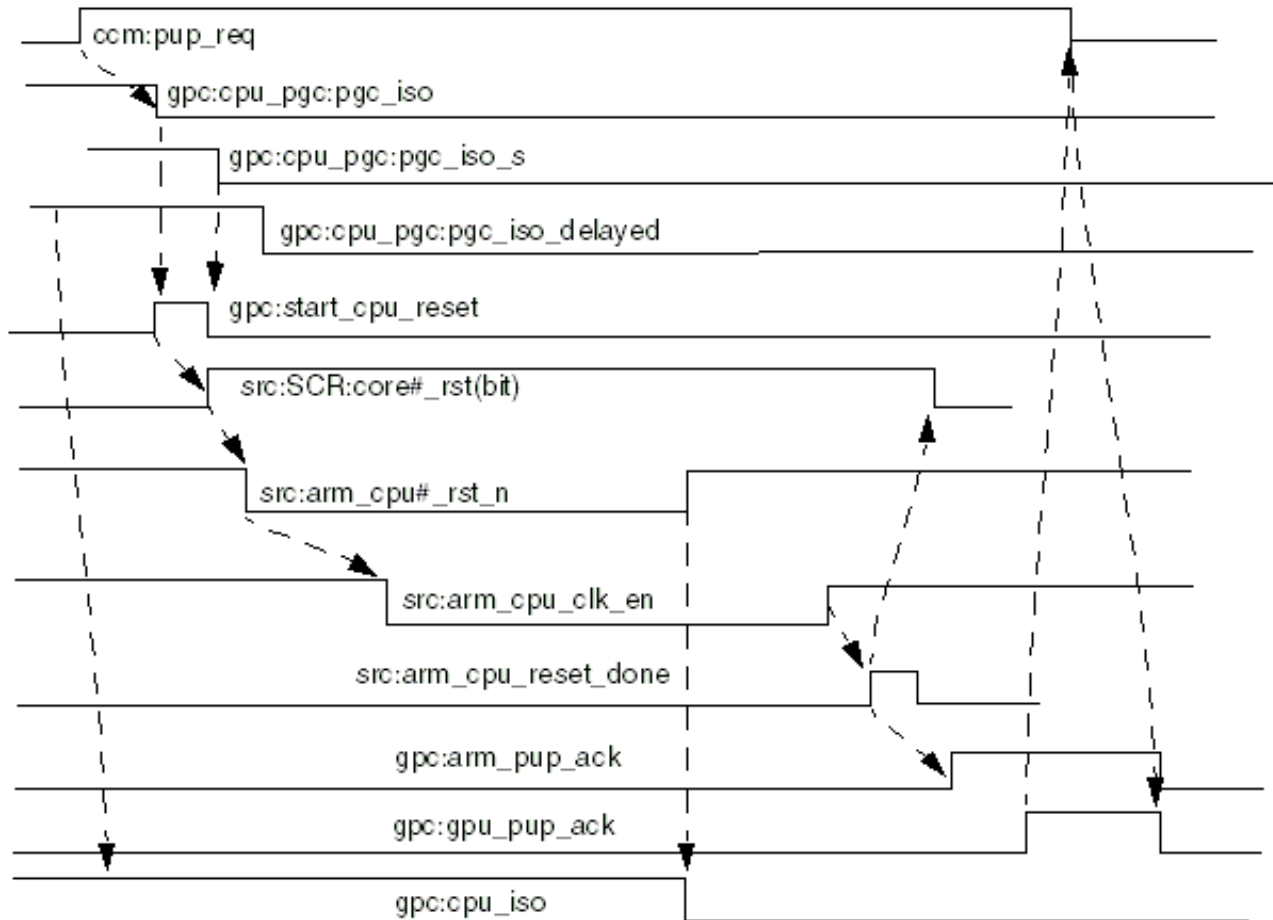
- CCM sends power down request when the chip is about to enter stop mode. The user should define which modules will be powered down (PGCR registers of corresponding PGC module, bit 0).

### PGC power up sequence:

- One of the power up irq is asserted.

- Power up request is asserted in GPC and in CCM.
- The Power Gated modules are powered up, according to PGC settings of appropriate module.

The Power Gated modules require reset after powering up. The next figure describes GPC-SRC handshake procedure for reset after power gating.



**Figure 25-15. GPC-SRC handshake for reset after power gating**

### 25.6.1 Overview

The Power Gating Controller (PGC) is a power management component that controls the power-down and power-up sequencing of individual subsystems.

The sequence timing is programmable using the PGC control registers. [Figure 25-16](#) shows PGC as part of the SoC's overall power management scheme.

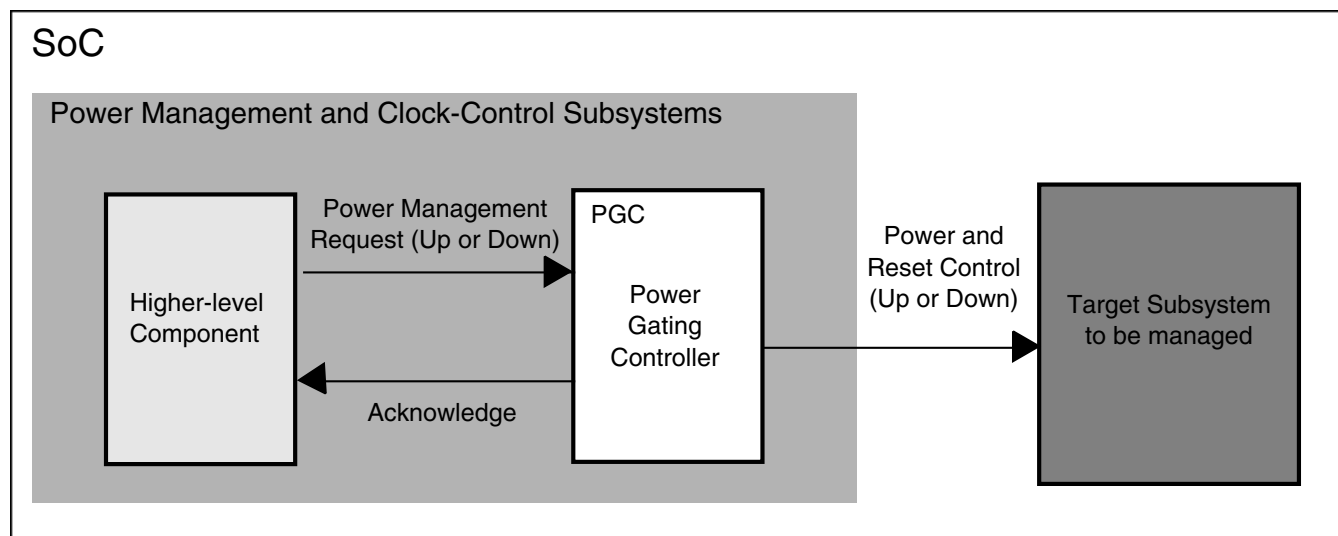


Figure 25-16. PGC Block Diagram

### 25.6.1.1 Features

Key features of the PGC include:

- Provides the ability to switch off power to a target subsystem.
- Generates power-up and power-down control sequences.
- Provides programmable registers to adjust the timing of the power control signals.

## 25.7 GPC Interrupt Controller (INTC)

The INTC (Interrupt Controller) detects an interrupt and generates the wakeup signal. It supports up to 128 interrupts.

### 25.7.1 Interrupt Controller features

The features of the GPC INTC are listed below.

Features:

- Supports up to 128 interrupts
- Provides an option to mask/unmask each interrupt
- Detects interrupts and generates the wake up signal
- 32-bits IP bus interface
- All registers are byte-accessible



## 25.8 GPC Memory Map/Register Definition

Detailed descriptions of each register can be found below.

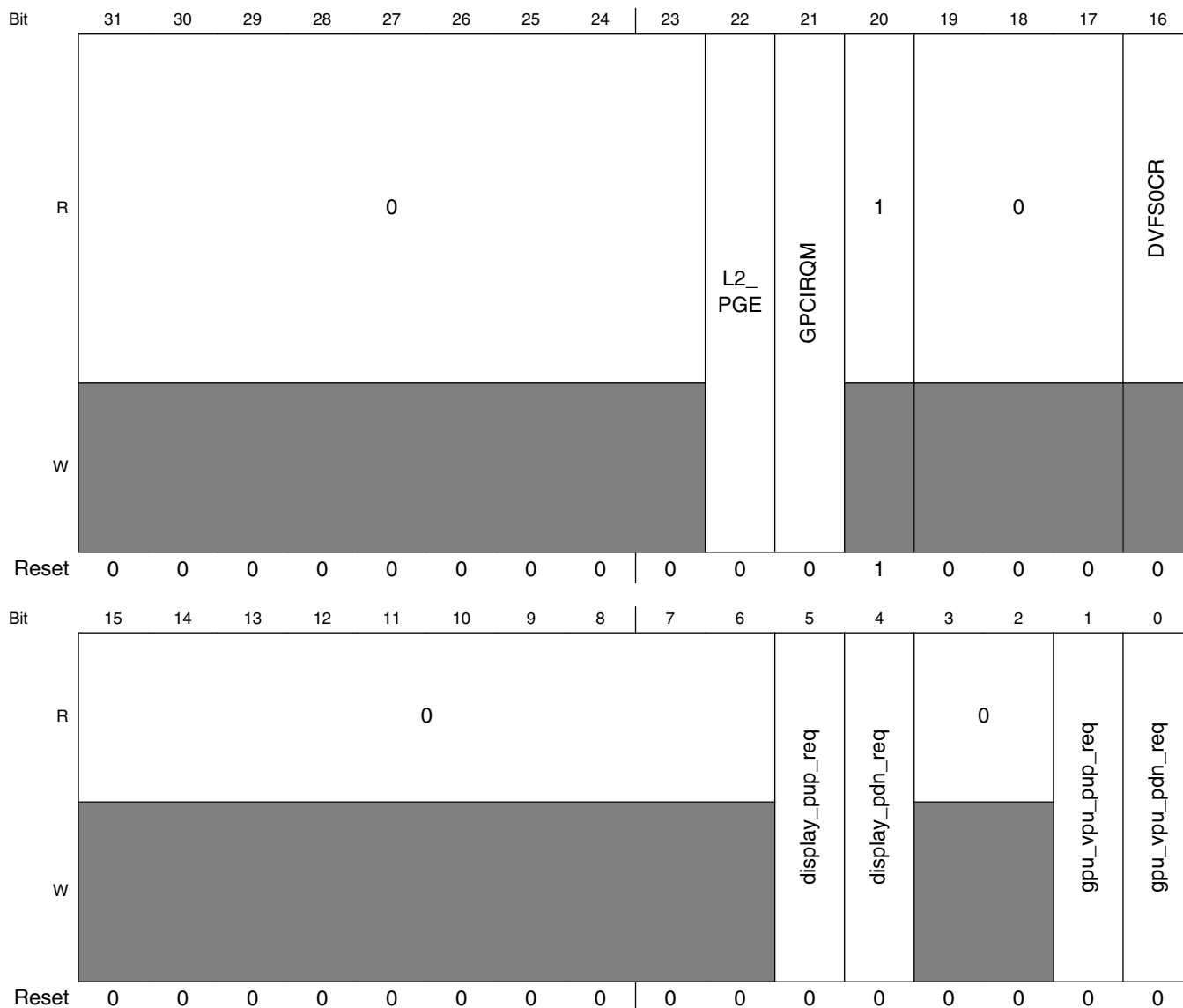
**GPC memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20D_C000	GPC Interface control register (GPC_CNTR)	32	R/W	0010_0000h	<a href="#">25.8.1/1125</a>
20D_C004	GPC Power Gating Register (GPC_PGR)	32	R/W	0000_0000h	<a href="#">25.8.2/1128</a>
20D_C008	IRQ masking register 1 (GPC_IMR1)	32	R/W	0000_0000h	<a href="#">25.8.3/1128</a>
20D_C00C	IRQ masking register 2 (GPC_IMR2)	32	R/W	0000_0000h	<a href="#">25.8.4/1129</a>
20D_C010	IRQ masking register 3 (GPC_IMR3)	32	R/W	0000_0000h	<a href="#">25.8.5/1129</a>
20D_C014	IRQ masking register 4 (GPC_IMR4)	32	R/W	0000_0000h	<a href="#">25.8.6/1130</a>
20D_C018	IRQ status resister 1 (GPC_ISR1)	32	R	0000_0000h	<a href="#">25.8.7/1130</a>
20D_C01C	IRQ status resister 2 (GPC_ISR2)	32	R	0000_0000h	<a href="#">25.8.8/1131</a>
20D_C020	IRQ status resister 3 (GPC_ISR3)	32	R	0000_0000h	<a href="#">25.8.9/1131</a>
20D_C024	IRQ status resister 4 (GPC_ISR4)	32	R	0000_0000h	<a href="#">25.8.10/1132</a>

### 25.8.1 GPC Interface control register (GPC\_CNTR)

## GPC Memory Map/Register Definition

Address: 20D\_C000h base + 0h offset = 20D\_C000h



### GPC\_CNTR field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 L2_PGE	L2 Cache Power Gate Enable 1 L2 cache power gate off request 0 no request
21 GPCIRQM	GPC interrupt/event masking 1 interrupt/event is masked 0 not masked
20 Reserved	This read-only field is reserved and always has the value 1.

Table continues on the next page...

**GPC\_CNTR field descriptions (continued)**

Field	Description
19–17 Reserved	This read-only field is reserved and always has the value 0.
16 DVFS0CR	DVFS0 (ARM) Change request (bit is read-only) 1 DVFS0 is requesting for frequency/voltage update 0 DVFS0 has no request
15–6 Reserved	This read-only field is reserved and always has the value 0.
5 display_pup_req	Display Power Up request. Self-cleared bit. <b>NOTE:</b> Software may directly control display power gate and utilize hardware control for reset sequence 1 Request Power Up sequence to start for Display 0 no request
4 display_pdn_req	Display Power Down request. Self-cleared bit. <b>NOTE:</b> Software may directly control display power gate and utilize hardware control for reset sequence 1 Request Power Down sequence to start for Display 0 no request
3–2 Reserved	This read-only field is reserved and always has the value 0.
1 gpu_vpu_pup_req	GPU Power Up request. Self-cleared bit. * Note: Power switch for GPU power domain is controlled by ANALOG configuration, not GPU PGC signals 1 Request Power Up sequence to start for GPU 0 no request
0 gpu_vpu_pdn_req	GPU Power Down request. Self-cleared bit. * Note: Power switch for GPU power domain is controlled by ANALOG configuration, not GPU PGC signals 1 Request Power Down sequence to start for GPU 0 no request

## 25.8.2 GPC Power Gating Register (GPC\_PGR)

Address: 20D\_C000h base + 4h offset = 20D\_C004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	DRCIC			0											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### GPC\_PGR field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–29 DRCIC	Debug ref cir in mux control  00 ccm_cosr_1_clk_in 01 ccm_cosr_2_clk_in 10 restricted 11 restricted
28–0 Reserved	This read-only field is reserved and always has the value 0.

## 25.8.3 IRQ masking register 1 (GPC\_IMR1)

IMR1 Register - masking of irq[63:32].

Address: 20D\_C000h base + 8h offset = 20D\_C008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IMR1																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### GPC\_IMR1 field descriptions

Field	Description
31–0 IMR1	IRQ[63:32] masking bits: 1-irq masked, 0-irq is not masked

## 25.8.4 IRQ masking register 2 (GPC\_IMR2)

IMR2 Register - masking of irq[95:64].

Address: 20D\_C000h base + Ch offset = 20D\_C00Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IMR2																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### GPC\_IMR2 field descriptions

Field	Description
31–0 IMR2	IRQ[95:64] masking bits: 1-irq masked, 0-irq is not masked

## 25.8.5 IRQ masking register 3 (GPC\_IMR3)

IMR3 Register - masking of irq[127:96].

Address: 20D\_C000h base + 10h offset = 20D\_C010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IMR3																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### GPC\_IMR3 field descriptions

Field	Description
31–0 IMR3	IRQ[127:96] masking bits: 1-irq masked, 0-irq is not masked

## 25.8.6 IRQ masking register 4 (GPC\_IMR4)

IMR4 Register - masking of irq[159:128].

Address: 20D\_C000h base + 14h offset = 20D\_C014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IMR4																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### GPC\_IMR4 field descriptions

Field	Description
31–0 IMR4	IRQ[159:128] masking bits: 1-irq masked, 0-irq is not masked

## 25.8.7 IRQ status resister 1 (GPC\_ISR1)

ISR1 Register - status of irq [63:32].

Address: 20D\_C000h base + 18h offset = 20D\_C018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISR1																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### GPC\_ISR1 field descriptions

Field	Description
31–0 ISR1	IRQ[63:32] status, read only

## 25.8.8 IRQ status resister 2 (GPC\_ISR2)

ISR2 Register - status of irq [95:64].

Address: 20D\_C000h base + 1Ch offset = 20D\_C01Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISR2																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**GPC\_ISR2 field descriptions**

Field	Description
31–0 ISR2	IRQ[95:64] status, read only

## 25.8.9 IRQ status resister 3 (GPC\_ISR3)

ISR3 Register - status of irq [127:96].

Address: 20D\_C000h base + 20h offset = 20D\_C020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISR3																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**GPC\_ISR3 field descriptions**

Field	Description
31–0 ISR3	IRQ[127:96] status, read only

## 25.8.10 IRQ status resister 4 (GPC\_ISR4)

ISR4 Register - status of irq [159:128].

Address: 20D\_C000h base + 24h offset = 20D\_C024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ISR4																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**GPC\_ISR4 field descriptions**

Field	Description
31–0 ISR4	IRQ[159:128] status, read only

## 25.9 PGC Memory Map/Register Definition

The PGC registers can be accessed only in supervisor mode.

Attempts to access registers when not in supervisor mode or attempts to access an unimplemented address location might trigger a bus transfer error. (The hardware asserts the signal `ips_xfr_err` if the PGC has been integrated with `resp_sel` tied low.) In this case, software should take appropriate action (such as ignore the error, log the error, or initiate a soft reset).

All PGC registers are byte-accessible.

### NOTE

The base address of each PGC module instantiation is specified in the GPC module. Absolute address values will be calculated by [GPC base address] + [PGC CPU/GPU/DISPLAY Offset].



## PGC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20D_C240	PGC Control Register (PGC_DISPLAY_CTRL)	32	R/W	0000_0000h	<a href="#">25.9.1/1133</a>
20D_C244	Power Up Sequence Control Register (PGC_DISPLAY_PUPSCR)	32	R/W	0000_0F01h	<a href="#">25.9.2/1134</a>
20D_C248	Pull Down Sequence Control Register (PGC_DISPLAY_PDNSCR)	32	R/W	0000_0101h	<a href="#">25.9.3/1135</a>
20D_C24C	Power Gating Controller Status Register (PGC_DISPLAY_SR)	32	R/W	0000_0000h	<a href="#">25.9.4/1135</a>
20D_C260	PGC Control Register (PGC_GPU_CTRL)	32	R/W	0000_0000h	<a href="#">25.9.5/1136</a>
20D_C264	Power Up Sequence Control Register (PGC_GPU_PUPSCR)	32	R/W	0000_0F01h	<a href="#">25.9.6/1137</a>
20D_C268	Pull Down Sequence Control Register (PGC_GPU_PDNSCR)	32	R/W	0000_0101h	<a href="#">25.9.7/1137</a>
20D_C26C	Power Gating Controller Status Register (PGC_GPU_SR)	32	R/W	0000_0000h	<a href="#">25.9.8/1138</a>
20D_C2A0	PGC Control Register (PGC_CPU_CTRL)	32	R/W	0000_0000h	<a href="#">25.9.9/1139</a>
20D_C2A4	Power Up Sequence Control Register (PGC_CPU_PUPSCR)	32	R/W	0000_0F01h	<a href="#">25.9.10/1139</a>
20D_C2A8	Pull Down Sequence Control Register (PGC_CPU_PDNSCR)	32	R/W	0000_0101h	<a href="#">25.9.11/1140</a>
20D_C2AC	Power Gating Controller Status Register (PGC_CPU_SR)	32	R/W	0000_0000h	<a href="#">25.9.12/1141</a>

## 25.9.1 PGC Control Register (PGC\_DISPLAY\_CTRL)

The PGCR enables the response to a power-down request.

Address: 20D\_C000h base + 240h offset = 20D\_C240h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## PGC\_DISPLAY\_CTRL field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

## PGC\_DISPLAY\_CTRL field descriptions (continued)

Field	Description
0 PCR	<p>Power Control</p> <p><b>NOTE:</b> PCR must not change from power-down request (pdn_req) assertion until the target subsystem is completely powered up.</p> <p>0 Do not switch off power even if pdn_req is asserted.</p> <p>1 Switch off power when pdn_req is asserted.</p>

## 25.9.2 Power Up Sequence Control Register (PGC\_DISPLAY\_PUPSCR)

The PUPSCR contains the power-up timing parameters.

Address: 20D\_C000h base + 244h offset = 20D\_C244h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																SW2ISO								0	SW						
W																	SW2ISO									SW						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	1

## PGC\_DISPLAY\_PUPSCR field descriptions

Field	Description
31–14 Reserved	This read-only field is reserved and always has the value 0.
13–8 SW2ISO	<p>After asserting power toggle on/off signal (switch_b), the PGC waits a number of clocks equal to the value of SW2ISO before negating isolation.</p> <p><b>NOTE:</b> SW2ISO must not be programmed to zero.</p>
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–0 SW	<p>After a power-up request (pup_req assertion), the PGC waits a number of clocks equal to the value of SW before asserting power toggle on/off signal (switch_b).</p> <p><b>NOTE:</b> SW must not be programmed to zero.</p> <p><b>NOTE:</b> The PGC clock is generated from the IPG_CLK_ROOT. for frequency configuration of the IPG_CLK_ROOT. See <a href="#">Clock Controller Module (CCM)</a>.</p>

### 25.9.3 Pull Down Sequence Control Register (PGC\_DISPLAY\_PDNSCR)

The PDNSCR contains the power-down timing parameters.

Address: 20D\_C000h base + 248h offset = 20D\_C248h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																ISO2SW				0		ISO									
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

#### PGC\_DISPLAY\_PDNSCR field descriptions

Field	Description
31–14 Reserved	This read-only field is reserved and always has the value 0.
13–8 ISO2SW	After asserting isolation, the PGC waits a number of clocks equal to the value of ISO2SW before negating power toggle on/off signal (switch_b). <b>NOTE:</b> ISO2SW must not be programmed to zero.
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–0 ISO	After a power-down request (pdn_req assertion), the PGC waits a number of clocks equal to the value of ISO before asserting isolation. <b>NOTE:</b> ISO must not be programmed to zero.

### 25.9.4 Power Gating Controller Status Register (PGC\_DISPLAY\_SR)

The PDNSCR contains the power-down timing parameters.

Address: 20D\_C000h base + 24Ch offset = 20D\_C24Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															PSR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## PGC\_DISPLAY\_SR field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 PSR	Power status. When in functional (or software-controlled debug) mode, PGC hardware sets PSR as soon as any of the power control output changes its state to one. Write one to clear this bit. Software should clear this bit after power up; otherwise, PSR continues to reflect the power status of the initial power down.  0 The target subsystem was not powered down for the previous power-down request. 1 The target subsystem was powered down for the previous power-down request.

## 25.9.5 PGC Control Register (PGC\_GPU\_CTRL)

The PGCR enables the response to a power-down request.

Address: 20D\_C000h base + 260h offset = 20D\_C260h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															PCR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## PGC\_GPU\_CTRL field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 PCR	Power Control  <b>NOTE:</b> PCR must not change from power-down request (pdn_req) assertion until the target subsystem is completely powered up.  0 Do not switch off power even if pdn_req is asserted. 1 Switch off power when pdn_req is asserted.

## 25.9.6 Power Up Sequence Control Register (PGC\_GPU\_PUPSCR)

The PUPSCR contains the power-up timing parameters.

Address: 20D\_C000h base + 264h offset = 20D\_C264h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														SW2ISO						0		SW									
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	1	

### PGC\_GPU\_PUPSCR field descriptions

Field	Description
31–14 Reserved	This read-only field is reserved and always has the value 0.
13–8 SW2ISO	After asserting power toggle on/off signal (switch_b), the PGC waits a number of clocks equal to the value of SW2ISO before negating isolation. <b>NOTE:</b> SW2ISO must not be programmed to zero.
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–0 SW	After a power-up request (pup_req assertion), the PGC waits a number of clocks equal to the value of SW before asserting power toggle on/off signal (switch_b). <b>NOTE:</b> SW must not be programmed to zero. <b>NOTE:</b> The PGC clock is generated from the IPG_CLK_ROOT. for frequency configuration of the IPG_CLK_ROOT. See <a href="#">Clock Controller Module (CCM)</a> .

## 25.9.7 Pull Down Sequence Control Register (PGC\_GPU\_PDNSCR)

The PDNSCR contains the power-down timing parameters.

Address: 20D\_C000h base + 268h offset = 20D\_C268h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																ISO2SW						0		ISO							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1

## PGC\_GPU\_PDNSCR field descriptions

Field	Description
31–14 Reserved	This read-only field is reserved and always has the value 0.
13–8 ISO2SW	After asserting isolation, the PGC waits a number of clocks equal to the value of ISO2SW before negating power toggle on/off signal (switch_b).  <b>NOTE:</b> ISO2SW must not be programmed to zero.
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–0 ISO	After a power-down request (pdn_req assertion), the PGC waits a number of clocks equal to the value of ISO before asserting isolation.  <b>NOTE:</b> ISO must not be programmed to zero.

## 25.9.8 Power Gating Controller Status Register (PGC\_GPU\_SR)

The PDNSCR contains the power-down timing parameters.

Address: 20D\_C000h base + 26Ch offset = 20D\_C26Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															PSR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## PGC\_GPU\_SR field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 PSR	Power status. When in functional (or software-controlled debug) mode, PGC hardware sets PSR as soon as any of the power control output changes its state to one. Write one to clear this bit. Software should clear this bit after power up; otherwise, PSR continues to reflect the power status of the initial power down.  0 The target subsystem was not powered down for the previous power-down request. 1 The target subsystem was powered down for the previous power-down request.

### 25.9.9 PGC Control Register (PGC\_CPU\_CTRL)

The PGCR enables the response to a power-down request.

Address: 20D\_C000h base + 2A0h offset = 20D\_C2A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															PCR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### PGC\_CPU\_CTRL field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 PCR	<p>Power Control</p> <p><b>NOTE:</b> PCR must not change from power-down request (pdn_req) assertion until the target subsystem is completely powered up.</p> <p>0 Do not switch off power even if pdn_req is asserted.</p> <p>1 Switch off power when pdn_req is asserted.</p>

### 25.9.10 Power Up Sequence Control Register (PGC\_CPU\_PUPSCR)

The PUPSCR contains the power-up timing parameters.

Address: 20D\_C000h base + 2A4h offset = 20D\_C2A4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																SW2ISO				0		SW									
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	1	

## PGC\_CPU\_PUPSCR field descriptions

Field	Description
31–14 Reserved	This read-only field is reserved and always has the value 0.
13–8 SW2ISO	After asserting , the PGC waits a number of clocks equal to the value of SW2ISO before negating isolation.  <b>NOTE:</b> SW2ISO must not be programmed to zero. The SW2ISO value should be chosen such that the delay before negating isolation is greater than the LDO ramp-up time.
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–0 SW	After a power-up request (pup_req assertion), the PGC waits a number of clocks equal to the value of SW before asserting .  <b>NOTE:</b> SW must not be programmed to zero.  <b>NOTE:</b> The PGC clock is generated from the IPG_CLK_ROOT. for frequency configuration of the IPG_CLK_ROOT. See <a href="#">Clock Controller Module (CCM)</a> .

## 25.9.11 Pull Down Sequence Control Register (PGC\_CPU\_PDNSCR)

The PDNSCR contains the power-down timing parameters.

Address: 20D\_C000h base + 2A8h offset = 20D\_C2A8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	0																ISO2SW								0	ISO							
W																	ISO2SW									ISO							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	

## PGC\_CPU\_PDNSCR field descriptions

Field	Description
31–14 Reserved	This read-only field is reserved and always has the value 0.
13–8 ISO2SW	After asserting isolation, the PGC waits a number of clocks equal to the value of ISO2SW before negating .  <b>NOTE:</b> ISO2SW must not be programmed to zero.
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–0 ISO	After a power-down request (pdn_req assertion), the PGC waits a number of clocks equal to the value of ISO before asserting isolation.  <b>NOTE:</b> ISO must not be programmed to zero.



## 25.9.12 Power Gating Controller Status Register (PGC\_CPU\_SR)

The PDNSCR contains the power-down timing parameters.

Address: 20D\_C000h base + 2ACh offset = 20D\_C2ACh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															PSR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### PGC\_CPU\_SR field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 PSR	Power status. When in functional (or software-controlled debug) mode, PGC hardware sets PSR as soon as any of the power control output changes its state to one. Write one to clear this bit. Software should clear this bit after power up; otherwise, PSR continues to reflect the power status of the initial power down.  0 The target subsystem was not powered down for the previous power-down request. 1 The target subsystem was powered down for the previous power-down request.

## 25.10 DVFSC Memory Map/Register Definition

### DVFSC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20D_C180	DVFS Thresholds (DVFSC_THRS)	32	R/W	0FAF_003Eh	<a href="#">25.10.1/1142</a>
20D_C184	DVFS Counters thresholds (DVFSC_COUN)	32	R/W	0007_0020h	<a href="#">25.10.2/1143</a>
20D_C188	DVFS general purpose bits weight (DVFSC_SIG1)	32	R/W	0000_0000h	<a href="#">25.10.3/1143</a>
20D_C18C	DVFS general purpose bits weight (DVFSC_DVFSSIG0)	32	R/W	0000_0000h	<a href="#">25.10.4/1144</a>
20D_C190	DVFS general purpose bit 0 weight counter (DVFSC_DVFSGPC0)	32	R/W	0000_0000h	<a href="#">25.10.5/1145</a>

Table continues on the next page...

## DVFS memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20D_C194	DVFS general purpose bit 1 weight counter (DVFS_C_DVFSGPC1)	32	R/W	0000_0000h	<a href="#">25.10.6/1146</a>
20D_C198	DVFS general purpose bits enables (DVFS_C_DVFSGPBT)	32	R/W	0000_0000h	<a href="#">25.10.7/1147</a>
20D_C19C	DVFS EMAC settings (DVFS_C_DVFSEMAC)	32	R/W	0000_0004h	<a href="#">25.10.8/1149</a>
20D_C1A0	DVFS Control (DVFS_C_CNTR)	32	R/W	0900_000Eh	<a href="#">25.10.9/1150</a>
20D_C1A4	DVFS Load Tracking Register 0, portion 0 (DVFS_C_DVFSLTR0_0)	32	R	0000_0000h	<a href="#">25.10.10/1153</a>
20D_C1A8	DVFS Load Tracking Register 0, portion 1 (DVFS_C_DVFSLTR0_1)	32	R	0000_0000h	<a href="#">25.10.11/1154</a>
20D_C1AC	DVFS Load Tracking Register 1, portion 0 (DVFS_C_DVFSLTR1_0)	32	R	0000_0000h	<a href="#">25.10.12/1155</a>
20D_C1B0	DVFS Load Tracking Register 3, portion 1 (DVFS_C_DVFSLTR1_1)	32	R	0000_0000h	<a href="#">25.10.13/1155</a>
20D_C1B4	DVFS pattern 0 length (DVFS_C_DVFSPT0)	32	R/W	0000_0010h	<a href="#">25.10.14/1156</a>
20D_C1B8	DVFS pattern 1 length (DVFS_C_DVFSPT1)	32	R/W	0000_0010h	<a href="#">25.10.15/1157</a>
20D_C1BC	DVFS pattern 2 length (DVFS_C_DVFSPT2)	32	R/W	0000_0010h	<a href="#">25.10.16/1157</a>
20D_C1C0	DVFS pattern 3 length (DVFS_C_DVFSPT3)	32	R/W	0000_0010h	<a href="#">25.10.17/1158</a>

## 25.10.1 DVFS Thresholds (DVFS\_C\_THRS)

Address: 20D\_C180h base + 0h offset = 20D\_C180h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																0															
W																																
Reset	0	0	0	0	1	1	1	1	1	0	1	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0

## DVFS\_C\_THRS field descriptions

Field	Description
31–28 Reserved	This read-only field is reserved and always has the value 0.
27–22 UPTHR	Upper threshold for load tracking
21–16 DWTHR	Down threshold for load tracking

Table continues on the next page...

**DVFSC\_THRS field descriptions (continued)**

Field	Description
15–6 Reserved	This read-only field is reserved and always has the value 0.
5–0 PNCTHR	Panic threshold for load tracking

**25.10.2 DVFS Counters thresholds (DVFSC\_COUN)**

Address: 20D\_C180h base + 4h offset = 20D\_C184h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	0								DN_CNT								0								UPCNT								
W																																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

**DVFSC\_COUN field descriptions**

Field	Description
31–24 Reserved	This read-only field is reserved and always has the value 0.
23–16 DN_CNT	Down counter threshold value
15–8 Reserved	This read-only field is reserved and always has the value 0.
7–0 UPCNT	UP counter threshold value

**25.10.3 DVFS general purpose bits weight (DVFSC\_SIG1)**

Address: 20D\_C180h base + 8h offset = 20D\_C188h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0							
R																																							
W	WSW15			WSW14			WSW13			WSW12			WSW11			WSW10			WSW9			WSW8			WSW7			WSW6			0								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0						

**DVFSC\_SIG1 field descriptions**

Field	Description
31–29 WSW15	General purpose load tracking signal weight dvfs_w_sig[15]
28–26 WSW14	General purpose load tracking signal weight dvfs_w_sig[14]

*Table continues on the next page...*

### DVFSC\_SIG1 field descriptions (continued)

Field	Description
25–23 WSW13	General purpose load tracking signal weight dvfs_w_sig[13]
22–20 WSW12	General purpose load tracking signal weight dvfs_w_sig[12]
19–17 WSW11	General purpose load tracking signal weight dvfs_w_sig[11]
16–14 WSW10	General purpose load tracking signal weight dvfs_w_sig[10]
13–11 WSW9	General purpose load tracking signal weight dvfs_w_sig[9]
10–8 WSW8	General purpose load tracking signal weight dvfs_w_sig[8]
7–5 WSW7	General purpose load tracking signal weight dvfs_w_sig[7]
4–2 WSW6	General purpose load tracking signal weight dvfs_w_sig[6]
1–0 Reserved	This read-only field is reserved and always has the value 0.

## 25.10.4 DVFS general purpose bits weight (DVFSC\_DVFSSIG0)

Address: 20D\_C180h base + Ch offset = 20D\_C18Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
R																	Reserved																			
W	WSW5				WSW4				WSW3				WSW2																				WSW1			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

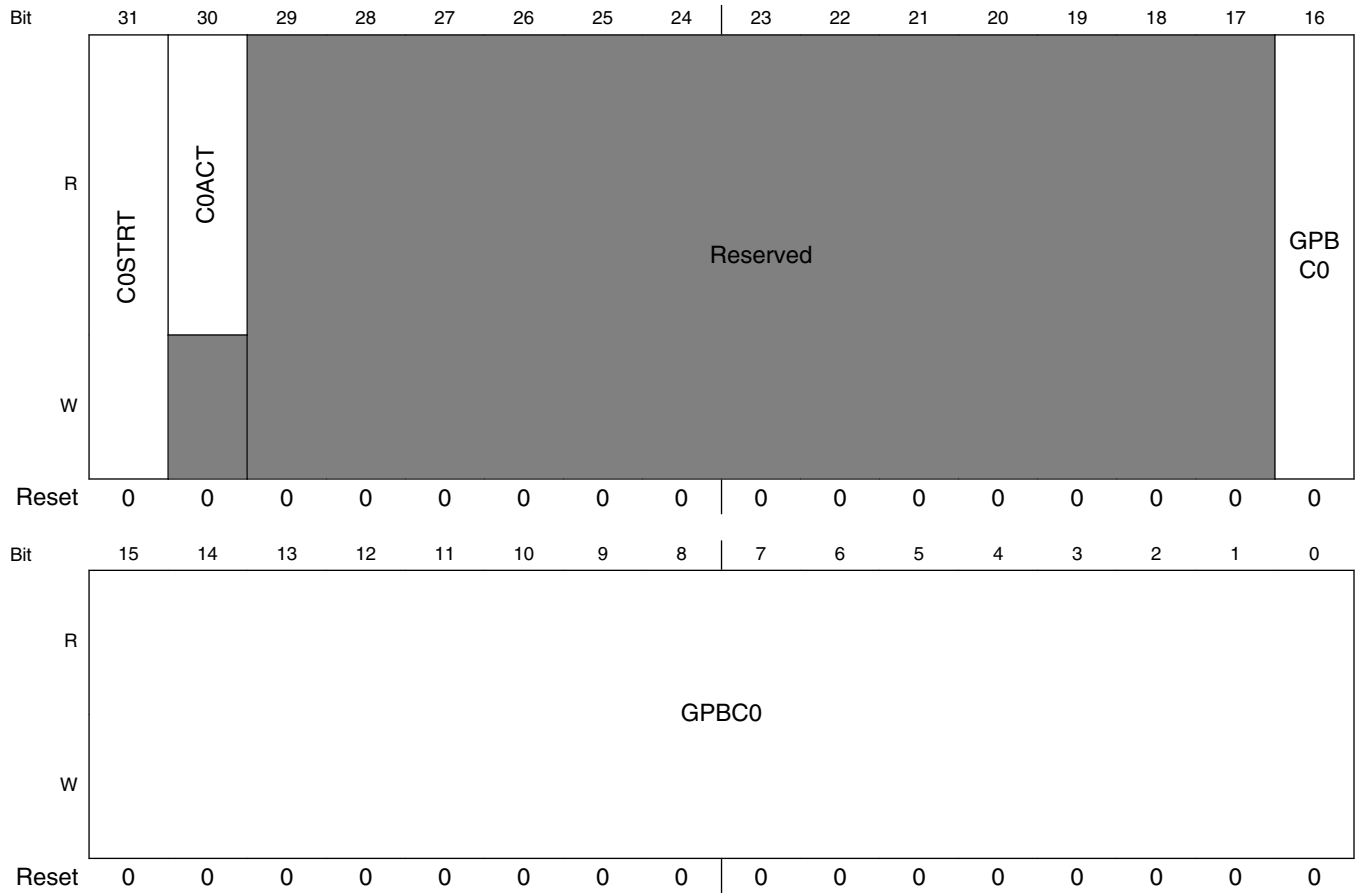
### DVFSC\_DVFSSIG0 field descriptions

Field	Description
31–29 WSW5	General purpose load tracking signal weight dvfs_w_sig[5]
28–26 WSW4	General purpose load tracking signal weight dvfs_w_sig[4]
25–23 WSW3	General purpose load tracking signal weight dvfs_w_sig[3]
22–20 WSW2	General purpose load tracking signal weight dvfs_w_sig[2]
19–12 -	This field is reserved. Reserved
11–6 WSW1	General purpose load tracking signal weight dvfs_w_sig[1]. This value is relevant during GPC1 counting period or when GPB1 is set.
5–0 WSW0	General purpose load tracking signal weight dvfs_w_sig[0]. This value is relevant during GPC0 counting period or when GPB0 is set.

## 25.10.5 DVFS general purpose bit 0 weight counter (DVFSC\_DVFSGPC0)

DVFS general purpose bits weight counter.

Address: 20D\_C180h base + 10h offset = 20D\_C190h



**DVFSC\_DVFSGPC0 field descriptions**

Field	Description
31 C0STRT	<p>C0STRT - Counter 0 start</p> <p>Setting of this bit will initialize down counting of the GPC0 value.</p> <p>Bit is self-cleared next cycle after setting.</p> <p>Any setting of this bit will re-start GPC0 counter to the GPC0 value.</p> <p>GPB0 bit disables (overrides) GPC0 counter - WSW0 weight is applicable continuously</p>
30 C0ACT	C0ACT - Counter 0 active indicator

*Table continues on the next page...*

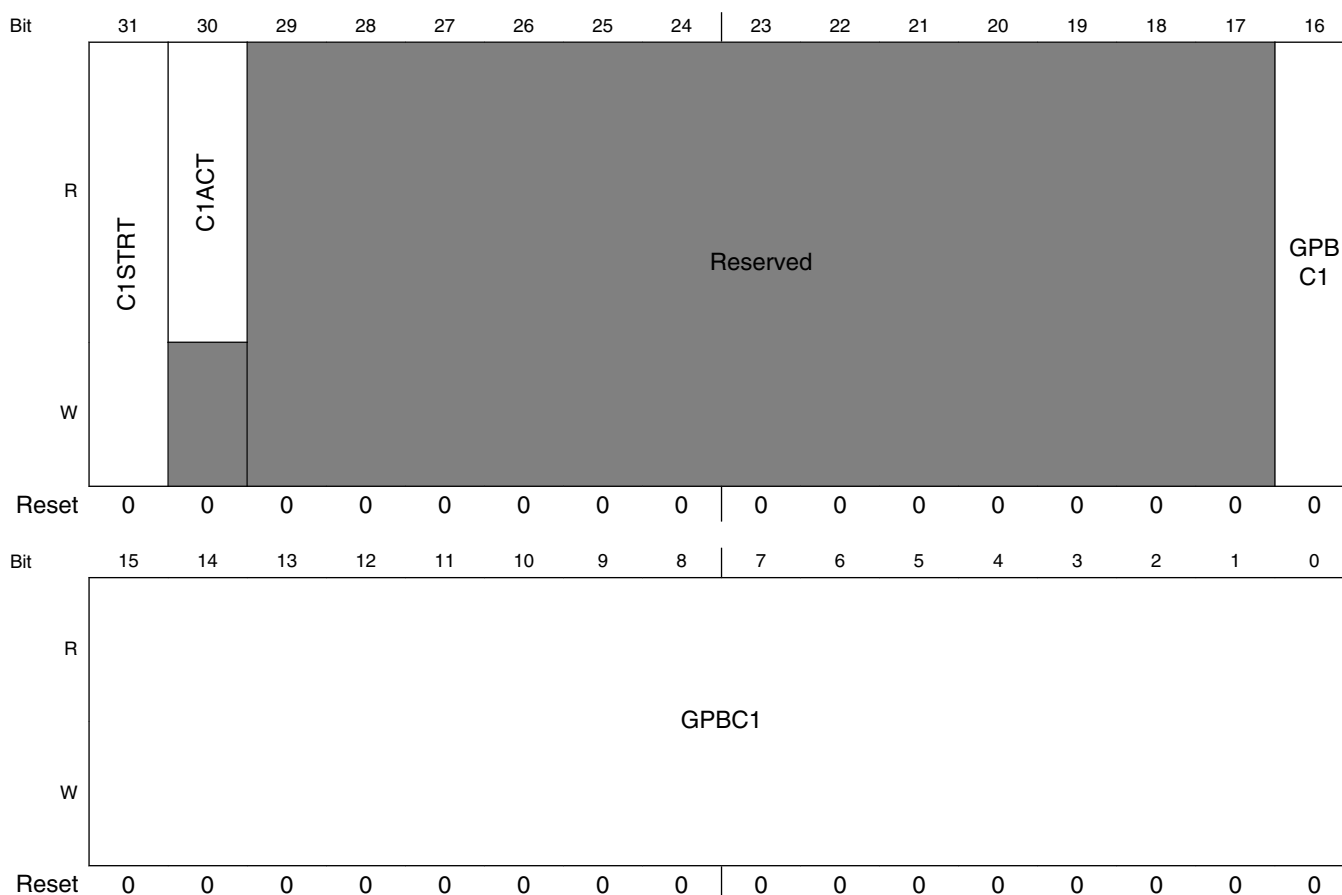
**DVFSC\_DVFSGPC0 field descriptions (continued)**

Field	Description
	1 General Purpose bit0 counter didn't reach value of "0" - the WSW0 is provided to DVFS calculation 0 General Purpose bit0 counter reached value of "0" - the instead of WSW0, "0" (zero) is provided to DVFS calculation
29–17 -	This field is reserved. reserved
16–0 GPBC0	GPBC0 - General Purpose bits Counter 0 During period of this counter the GeP bit 0 will be set and WSW0 will be added to the calculations.

**25.10.6 DVFS general purpose bit 1 weight counter (DVFSC\_DVFSGPC1)**

DVFS general purpose bits weight counter1.

Address: 20D\_C180h base + 14h offset = 20D\_C194h



**DVFSC\_DVFSGPC1 field descriptions**

Field	Description
31 C1STRT	C1STRT - Counter 1start Setting of this bit will initialize down counting of the GPC1 value. Bit is self-cleared next cycle after setting. Any setting of this bit will re-start GPC1 counter to the GPC1 value. GPB1 bit disables (overrides) GPC1 counter - WSW1 weight is applicable continuously
30 C1ACT	C1ACT - Counter 1 active indicator 1 General Purpose bit1 counter didn't reach value of "0" - the WSW1 is provided to DVFS calculation 0 General Purpose bit1 counter reached value of "0" - the instead of WSW1, "0" (zero) is provided to DVFS calculation
29–17 -	This field is reserved. reserved
16–0 GPBC1	GPBC1 - General Purpose bits Counter 1 During period of this counter the GeP bit 1 will be set and WSW1 will be added to the calculations.

**25.10.7 DVFS general purpose bits enables (DVFSC\_DVFSGPBT)**

Address: 20D\_C180h base + 18h offset = 20D\_C198h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPB15	GPB14	GPB13	GPB12	GPB11	GPB10	GPB9	GPB8	GPB7	GPB6	GPB5	GPB4	GPB3	GPB2	GPB1	GPB0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DVFSC\_DVFSGPBT field descriptions**

Field	Description
31–16 -	This field is reserved. reserved
15 GPB15	General purpose bit 15. Its weight is set by WSW15 value.
14 GPB14	General purpose bit 14. Its weight is set by WSW14 value.

*Table continues on the next page...*

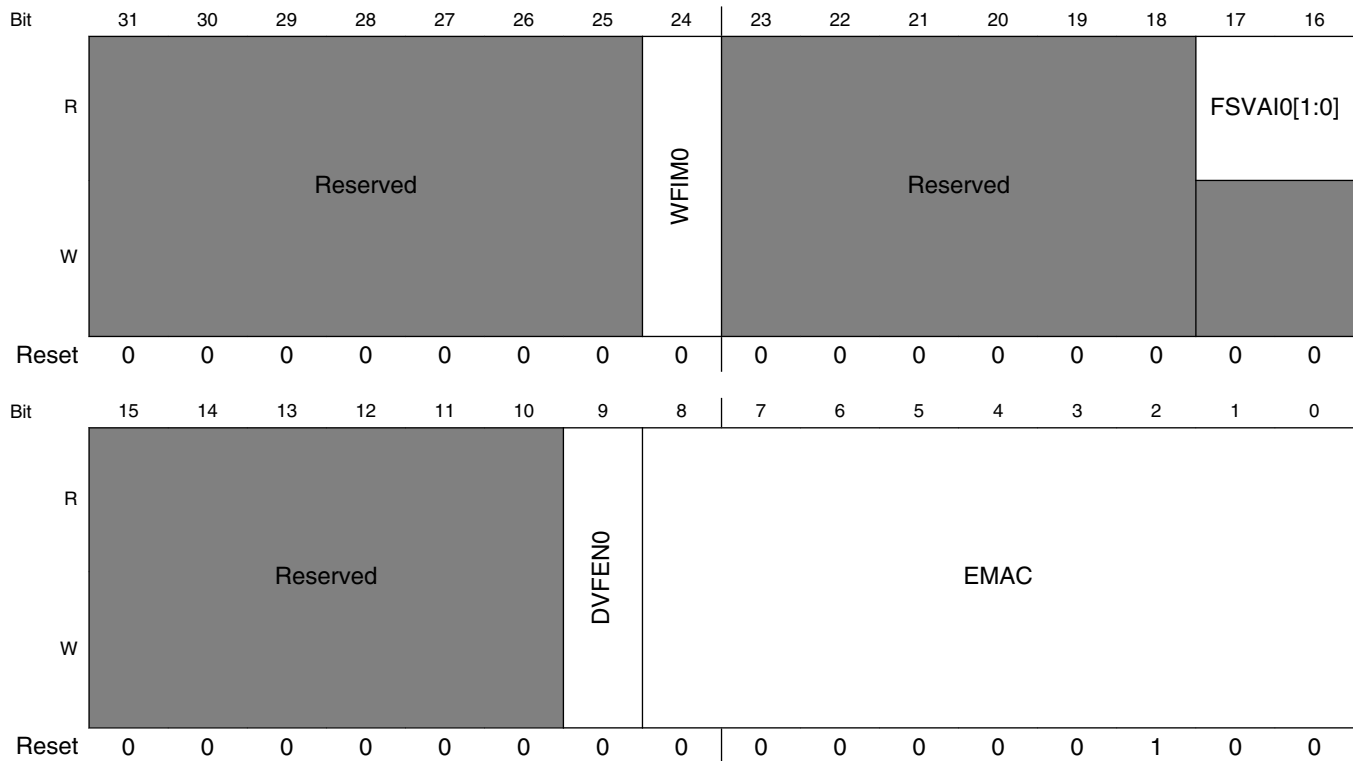
**DVFSC\_DVFSGPBT field descriptions (continued)**

<b>Field</b>	<b>Description</b>
13 GPB13	General purpose bit 13. Its weight is set by WSW13 value.
12 GPB12	General purpose bit 12. Its weight is set by WSW12 value.
11 GPB11	General purpose bit 11. Its weight is set by WSW11 value.
10 GPB10	General purpose bit 10. Its weight is set by WSW10 value.
9 GPB9	General purpose bit 9. Its weight is set by WSW9 value.
8 GPB8	General purpose bit 8. Its weight is set by WSW8 value.
7 GPB7	General purpose bit 7. Its weight is set by WSW7 value.
6 GPB6	General purpose bit 6. Its weight is set by WSW6 value.
5 GPB5	General purpose bit 5. Its weight is set by WSW5 value.
4 GPB4	General purpose bit 4. Its weight is set by WSW4 value.
3 GPB3	General purpose bit 3. Its weight is set by WSW3 value.
2 GPB2	General purpose bit 2. Its weight is set by WSW2 value.
1 GPB1	General purpose bit 1. Its weight is set by WSW1 value. IF set (1), the GPBC1 operation is disregarded, WSW1 value is applied continuously.
0 GPB0	General purpose bit 0. Its weight is set by WSW0 value. IF set (1), the GPBC0 operation is disregarded, WSW0 value is applied continuously.



## 25.10.8 DVFS EMAC settings (DVFSC\_DVFSEMAC)

Address: 20D\_C180h base + 1Ch offset = 20D\_C19Ch



**DVFSC\_DVFSEMAC field descriptions**

Field	Description
31–25 -	This field is reserved. Reserved
24 WFIM0	DVFS Wait for Interrupt of core 0 mask bit  0 Wait for interrupt of core 0 isn't masked 1 Wait for interrupt of core 0 is masked.
23–18 -	This field is reserved. Reserved
17–16 FSVAI0[1:0]	DVFS Frequency adjustment status of core 0. These status bits indicate that frequency should be changed, following load of core 0.  00 no change 01 frequency should be increased. Low priority source for interrupt. Interrupt is asserted, if FSVAIM=0. Interrupt is masked if MAXF = 1 (highest frequency). 10 frequency should be decreased. Interrupt is asserted, if FSVAIM=0. Interrupt is masked if MINF= 1 (lowest frequency). 11 frequency should be increased immediately. High priority source of interrupt. Interrupt is asserted, if FSVAIM=0. Interrupt is masked if MAXF = 1 (highest frequency).

Table continues on the next page...

**DVFSC\_DVFSEMAC field descriptions (continued)**

Field	Description
15–10 -	This field is reserved. Reserved
9 DVFEN0	DVFS tracking for core0 enable.  1 DVFS enabled. 0 DVFS disabled.
8–0 EMAC	EMAC - EMA control value

**25.10.9 DVFS Control (DVFSC\_CNTR)****Table 25-39. DIV3CK division**

DIV3CK setting	dividing ratio	sum_3 passing bits	div_1_clk cumulative divider
00	1	4-0	$1 \times 512 = 512$
001	4	6-2	$4 \times 512 = 2048$
010	16	8-4	$16 \times 512 = 8192$
011	64	10-6	$64 \times 512 = 32768$
100	256	12-8	$256 \times 512 = 131072$
101	1024	16-10	$1024 \times 512 = 524288$

**Table 25-40. Preliminary Divider definition**

DIV_RATIO value	ARM clk division ratio
000000	1
000001	2
000010	3
...	...

Address: 20D\_C180h base + 20h offset = 20D\_C1A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DIV3CK			DVFEV	LBMI	LBFL1	LBFL0	DVFIS	PIRQS	FSVAIM	FSVAI[1:0]		0	MAXF	MINF	DIV_RATIO
W																
Reset	0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DIV_RATIO					0	PFUE	PFUS			LTBRSH	LTBRSR		0		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0

**DVFSC\_CNTR field descriptions**

Field	Description
31–29 DIV3CK	DIV3CK - div_3_clk division ratio inside the DVFS module. According to the <a href="#">Table 25-39</a>
28 DVFEV	Always give a DVFS event. 0 Do not give an event always. 1 Always give event.
27 LBMI	Load buffer full mask interrupt. This bit masks the generation of this interrupt. Load buffer full interrupt is masked (LBFL0 and LBFL1 bits still will be updated, but interrupt won't be generated) Load buffer full interrupt is enabled.
26 LBFL1	Load buffer 1 - full status bit. This bit indicates that log buffer registers are full. The bit is set to 1 automatically. An interrupt will be generated if LBMI bit is set to "0" Write '1' to clear. (write '0' leaves bit unchanged) 1 Load buffer0 is full. 0 Load buffer0 is not full.
25 LBFL0	Load buffer 0 - full status bit. This bit indicates that log buffer registers are full. The bit is set to 1 automatically. An interrupt will be generated if LBMI bit is set to "0" Write '1' to clear. (write '0' leaves bit unchanged)

*Table continues on the next page...*

**DVFS\_CNTR field descriptions (continued)**

Field	Description
	1 Load buffer1 is full. 0 Load buffer1 is not full.
24 DVFS	DVFS Interrupt select. These bits define destination of DVFS interrupts.  1 MCU interrupt will be generated for DVFS events. 0 SDMA interrupt will be generated for DVFS events.
23 PIRQS	PIRQS - Pattern IRQ Source * write '1' to clear. Writing '1' will clear interrupt if interrupt was from pattern  1 DVFS IRQ source was from pattern 0 DVFS IRQ source was not from pattern
22 FSVAIM	DVFS Frequency adjustment interrupt mask. This bit masks the DVFS frequency adjustment interrupt. FSVAI status bits will be still asserted in relevant cases.  1 interrupt is masked. 0 interrupt is enabled.
21–20 FSVAI[1:0]	FSVAI DVFS Frequency adjustment interrupt. These status bits indicate that the system frequency should be changed.  00 no interrupt 01 frequency should be increased. Low priority interrupt. Interrupt is asserted, if FSVAIM=0. Interrupt is masked if MAXF = 1 (highest frequency). 10 frequency should be decreased. Interrupt is asserted, if FSVAIM=0. Interrupt is masked if MINF= 1 (lowest frequency). 11 frequency should be increased immediately. High priority interrupt. Interrupt is asserted, if FSVAIM=0. Interrupt is masked if MAXF = 1 (highest frequency).
19 Reserved	This read-only field is reserved and always has the value 0.
18 MAXF	Maximum frequency reached. Interrupt will not be created in maximum frequency reached and frequency increase required.  1 max frequency reached 0 max frequency not reached
17 MINF	Minimum frequency reached. Interrupt will not be created in minimum frequency reached and frequency decrease required.  1 min frequency reached 0 min frequency not reached
16–11 DIV_RATIO	DIV_RATIO - Divider value. Divider divides the input ARM clock, following the table <a href="#">Table 25-40</a>
10 Reserved	This read-only field is reserved and always has the value 0.
9 PFUE	PFUE - Period Frequency Update Enable  1 enabled 0 disabled
8–6 PFUS	PFUS - Periodic Frequency Update Status

*Table continues on the next page...*

**DVFSC\_CNTR field descriptions (continued)**

Field	Description
	000 no update 100 DVFSPT0 period, previous finished(can be performance level decrease) 101 DVFSPT1 period, previous finished(can be EMA-detected performance level) 110 DVFSPT2 period, previous finished(can be performance level increase) 111 DVFSPT3 period, previous finished (can be EMA-detected performance level)
5 LTBRSH	LTBRSH - Load Tracking Buffer Register Shift: 0 values of [5:2] of the selected input are saving in Load Tracking Buffer 1 values of [4:1] of the selected input are saving in Load Tracking Buffer
4-3 LTBRSR	LTBRSR - Load Tracking Buffer Register Source: 00 pre_Id_add 01 Id_add 10 ema_Id 11 reserved
2-0 Reserved	This read-only field is reserved and always has the value 0.

### 25.10.10 DVFS Load Tracking Register 0, portion 0 (DVFSC\_DVFSLTR0\_0)

Address: 20D\_C180h base + 24h offset = 20D\_C1A4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LTS0_7				LTS0_6				LTS0_5				LTS0_4				LTS0_3				LTS0_2				LTS0_1				LTS0_0			
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**DVFSC\_DVFSLTR0\_0 field descriptions**

Field	Description
31-28 LTS0_7	core 0 Load Tracking Sample 7
27-24 LTS0_6	core 0 Load Tracking Sample 6
23-20 LTS0_5	core 0 Load Tracking Sample 5
19-16 LTS0_4	core 0 Load Tracking Sample 4
15-12 LTS0_3	core 0 Load Tracking Sample 3
11-8 LTS0_2	core 0 Load Tracking Sample 2

Table continues on the next page...

**DVFS\_DVFSLTR0\_0 field descriptions (continued)**

Field	Description
7–4 LTS0_1	core 0 Load Tracking Sample 1
3–0 LTS0_0	core 0 Load Tracking Sample 0

**25.10.11 DVFS Load Tracking Register 0, portion 1 (DVFS\_DVFSLTR0\_1)**

Address: 20D\_C180h base + 28h offset = 20D\_C1A8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LTS0_15				LTS0_14				LTS0_13				LTS0_12				LTS0_11				LTS0_10				LTS0_9				LTS0_8			
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**DVFS\_DVFSLTR0\_1 field descriptions**

Field	Description
31–28 LTS0_15	core 0 Load Tracking Sample 15
27–24 LTS0_14	core 0 Load Tracking Sample 14
23–20 LTS0_13	core 0 Load Tracking Sample 13
19–16 LTS0_12	core 0 Load Tracking Sample 12
15–12 LTS0_11	core 0 Load Tracking Sample 11
11–8 LTS0_10	core 0 Load Tracking Sample 10
7–4 LTS0_9	core 0 Load Tracking Sample 9
3–0 LTS0_8	core 0 Load Tracking Sample 8

## 25.10.12 DVFS Load Tracking Register 1, portion 0 (DVFSC\_DVFSLTR1\_0)

Address: 20D\_C180h base + 2Ch offset = 20D\_C1ACh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LTS1_7				LTS1_6				LTS1_5				LTS1_4				LTS1_3				LTS1_2				LTS1_1				LTS1_0			
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### DVFSC\_DVFSLTR1\_0 field descriptions

Field	Description
31–28 LTS1_7	core 0 Load Tracking Sample 7
27–24 LTS1_6	core 0 Load Tracking Sample 6
23–20 LTS1_5	core 0 Load Tracking Sample 5
19–16 LTS1_4	core 0 Load Tracking Sample 4
15–12 LTS1_3	core 0 Load Tracking Sample 3
11–8 LTS1_2	core 0 Load Tracking Sample 2
7–4 LTS1_1	core 0 Load Tracking Sample 1
3–0 LTS1_0	core 0 Load Tracking Sample 0

## 25.10.13 DVFS Load Tracking Register 3, portion 1 (DVFSC\_DVFSLTR1\_1)

Address: 20D\_C180h base + 30h offset = 20D\_C1B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LTS1_15				LTS1_14				LTS1_13				LTS1_12				LTS1_11				LTS1_10				LTS1_9				LTS1_8			
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

## DVFS\_DVFSLTR1\_1 field descriptions

Field	Description
31–28 LTS1_15	core 0 Load Tracking Sample 15
27–24 LTS1_14	core 0 Load Tracking Sample 14
23–20 LTS1_13	core 0 Load Tracking Sample 13
19–16 LTS1_12	core 0 Load Tracking Sample 12
15–12 LTS1_11	core 0 Load Tracking Sample 11
11–8 LTS1_10	core 0 Load Tracking Sample 10
7–4 LTS1_9	core 0 Load Tracking Sample 9
3–0 LTS1_8	core 0 Load Tracking Sample 8

## 25.10.14 DVFS pattern 0 length (DVFS\_DVFSPT0)

Address: 20D\_C180h base + 34h offset = 20D\_C1B4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved														PT0A	FPTN
W																0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FPTN0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

## DVFS\_DVFSPT0 field descriptions

Field	Description
31–18 -	This field is reserved. reserved
17 PT0A	PT0A - Pattern 0 currently active (read-only)  1 active 0 non-active
16–0 FPTN0	FPTN0 - Frequency pattern 0 counter During period of this counter the frequency will be reduced from the EMA-detected level.



## 25.10.15 DVFS pattern 1 length (DVFSC\_DVFSPT1)

Address: 20D\_C180h base + 38h offset = 20D\_C1B8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved														PT1A	FPTN
W																1
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FPTN1															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

**DVFSC\_DVFSPT1 field descriptions**

Field	Description
31–18 -	This field is reserved. reserved
17 PT1A	PT1A - Pattern 1 currently active (read-only)  1 active 0 non-active
16–0 FPTN1	FPTN1 - Frequency pattern 1 counter During period of this counter the frequency will be set to the EMA-detected level.

## 25.10.16 DVFS pattern 2 length (DVFSC\_DVFSPT2)

Address: 20D\_C180h base + 3Ch offset = 20D\_C1BCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	P2THR							Reserved							PT2A	FPTN
W																2
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FPTN2															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

**DVFSC\_DVFSPT2 field descriptions**

Field	Description
31–26 P2THR	P2THR - Pattern 2 Threshold

*Table continues on the next page...*

### DVFSC\_DVFSPT2 field descriptions (continued)

Field	Description
	Threshold of current DVFS load (after EMA), for generating interrupts with PFUS indicators 110, 111. If the current performance is greater than the P2THR value, the interrupts will be generated. Otherwise, pattern delay will be counted, but without interrupt generation.
25–18 -	This field is reserved. reserved
17 PT2A	PT2A - Pattern 2 currently active (read-only)  1 active 0 non-active
16–0 FPTN2	FPTN2 - Frequency pattern 2 counter  During period of this counter the frequency will be increased to higher, than detected by the EMA-detected level.

### 25.10.17 DVFS pattern 3 length (DVFSC\_DVFSPT3)

Address: 20D\_C180h base + 40h offset = 20D\_C1C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved														PT3A	FPTN
W	Reserved															3
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FPTN3															
W	FPTN3															
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

### DVFSC\_DVFSPT3 field descriptions

Field	Description
31–18 -	This field is reserved. reserved
17 PT3A	PT3A - Pattern 3 currently active (read-only)  1 active 0 non-active
16–0 FPTN3	FPTN3 - Frequency pattern 3 counter  During period of this counter the frequency will be set to the EMA-detected level.

# Chapter 26

## General Purpose Input/Output (GPIO)

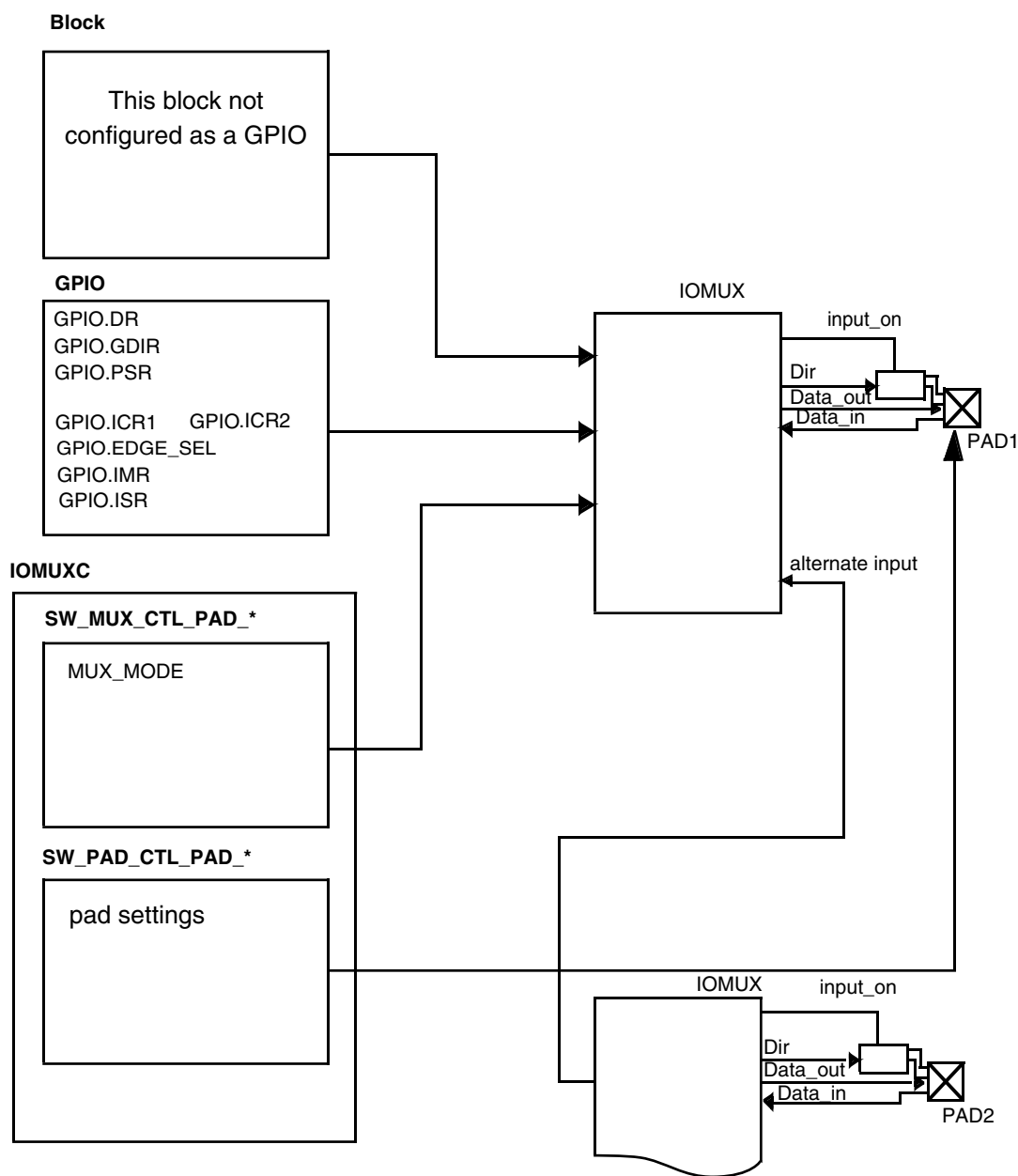
### 26.1 Overview

The GPIO general-purpose input/output peripheral provides dedicated general-purpose pins that can be configured as either inputs or outputs.

When configured as an output, it is possible to write to an internal register to control the state driven on the output pin. When configured as an input, it is possible to detect the state of the input by reading the state of an internal register. In addition, the GPIO peripheral can produce CORE interrupts.

The GPIO is one of the blocks controlling the IOMUX of the chip.

[Figure 26-1](#) shows the chip multiplexing scheme.



**Figure 26-1. Chip IOMUX Scheme**

The GPIO functionality is provided through eight registers, an edge-detect circuit, and interrupt generation logic.

The eight registers are:

- Data register (GPIO\_DR)
- GPIO direction register (GPIO\_GDIR)
- Pad sample register (GPIO\_PSR)
- Interrupt control registers (GPIO\_ICR1, GPIO\_ICR2)

- Edge select register (GPIO\_EDGE\_SEL)
- Interrupt mask register (GPIO\_IMR)
- Interrupt status register (GPIO\_ISR)

These registers are described in detail in [GPIO Memory Map/Register Definition](#).

Each GPIO input has a dedicated edge-detect circuit which can be configured through software to detect rising edges, falling edges, logic low-levels or logic high-levels on the input signals. The outputs of the edge detect circuits are optionally masked by setting the corresponding bit in the interrupt mask register (GPIO\_IMR). These qualified outputs are OR'ed together to generate two one-bit interrupt lines:

- Combined interrupt indication for GPIOx signals 0 - 15
- Combined interrupt indication for GPIOx signals 16 - 31

In addition, GPIO1 provides visibility to each of its 8 low order interrupt sources (i.e. GPIO1 interrupt n, for  $n = 0 - 7$ ). However, individual interrupt indications from other GPIOx are not available.

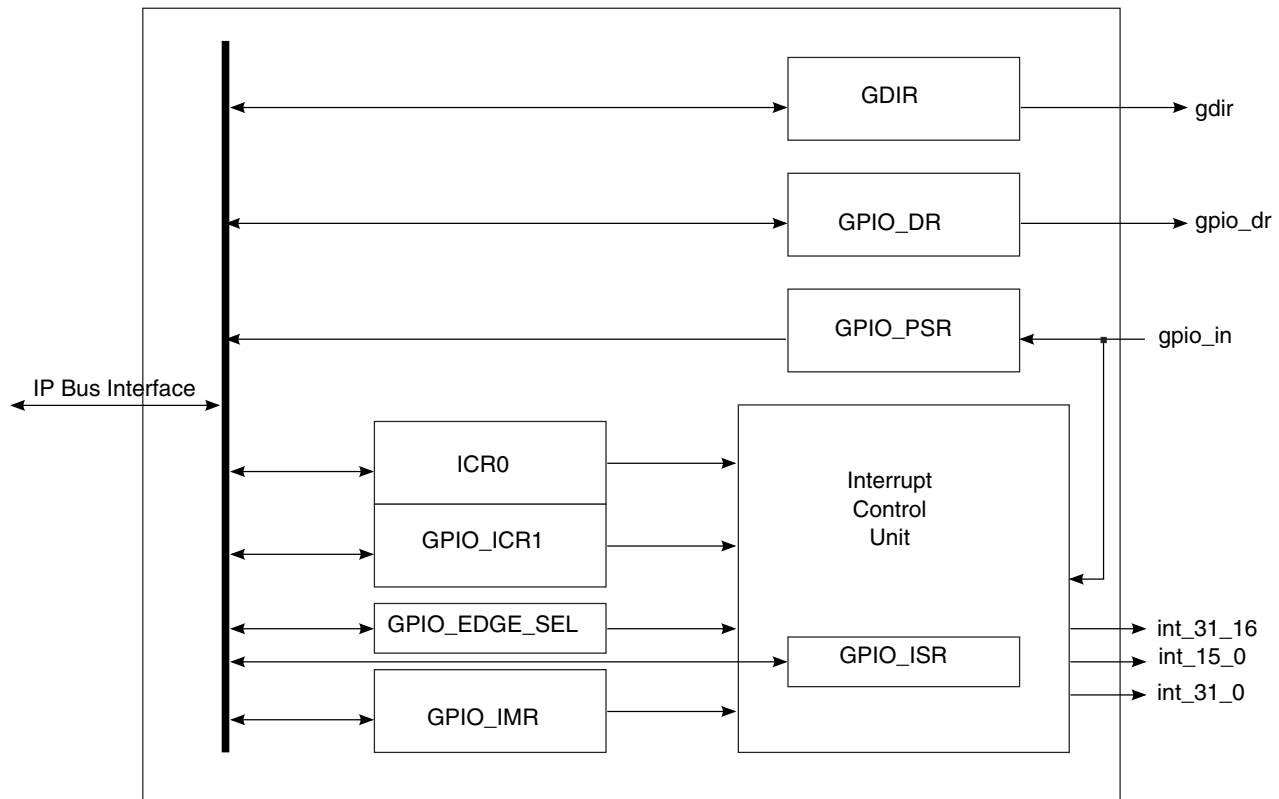
The GPIO edge detection is described further in [Interrupt Control Unit](#).

The GPIO's overall functionality is described further in [GPIO Functional Description](#).

## 26.1.1 Block Diagram

The GPIO subsystem contains 8 GPIO blocks which can generate and control up to 32 signals for general purpose.

A block diagram of the GPIO is shown in [Figure 26-2](#).



**Figure 26-2. GPIO Block Diagram**

## 26.1.2 Features

The GPIO includes the following features:

- General purpose input/output logic capabilities:
  - Drives specific data to output using the data register (GPIO\_DR)
  - Controls the direction of the signal using the GPIO direction register (GPIO\_GDIR)
  - Enables the core to sample the status of the corresponding inputs by reading the pad sample register (GPIO\_PSR).
- GPIO interrupt capabilities:
  - Supports up to 32 interrupts

- Identifies interrupt edges
- Generates three active-high interrupts to the SoC interrupt controller

## 26.2 External Signals

The tables found here describe the external signals of GPIO.

**Table 26-1. GPIO1 External Signals**

Signal	Description	Pad	Mode	Direction
GPIO1_IO00	-	AUD_RXFS	ALT5	IO
GPIO1_IO01	-	AUD_RXC	ALT5	IO
GPIO1_IO02	-	AUD_RXD	ALT5	IO
GPIO1_IO03	-	AUD_TXC	ALT5	IO
GPIO1_IO04	-	AUD_TXFS	ALT5	IO
GPIO1_IO05	-	AUD_TXD	ALT5	IO
GPIO1_IO06	-	AUD_MCLK	ALT5	IO
GPIO1_IO07	-	EPDC_D0	ALT5	IO
GPIO1_IO08	-	EPDC_D1	ALT5	IO
GPIO1_IO09	-	EPDC_D2	ALT5	IO
GPIO1_IO10	-	EPDC_D3	ALT5	IO
GPIO1_IO11	-	EPDC_D4	ALT5	IO
GPIO1_IO12	-	EPDC_D5	ALT5	IO
GPIO1_IO13	-	EPDC_D6	ALT5	IO
GPIO1_IO14	-	EPDC_D7	ALT5	IO
GPIO1_IO15	-	EPDC_D8	ALT5	IO
GPIO1_IO16	-	EPDC_D9	ALT5	IO
GPIO1_IO17	-	EPDC_D10	ALT5	IO
GPIO1_IO18	-	EPDC_D11	ALT5	IO
GPIO1_IO19	-	EPDC_D12	ALT5	IO
GPIO1_IO20	-	EPDC_D13	ALT5	IO
GPIO1_IO21	-	EPDC_D14	ALT5	IO
GPIO1_IO22	-	EPDC_D15	ALT5	IO
GPIO1_IO23	-	EPDC_SDCLK	ALT5	IO
GPIO1_IO24	-	EPDC_SDLE	ALT5	IO
GPIO1_IO25	-	EPDC_SDOE	ALT5	IO
GPIO1_IO26	-	EPDC_SDSHR	ALT5	IO
GPIO1_IO27	-	EPDC_SDCE0	ALT5	IO
GPIO1_IO28	-	EPDC_SDCE1	ALT5	IO
GPIO1_IO29	-	EPDC_SDCE2	ALT5	IO
GPIO1_IO30	-	EPDC_SDCE3	ALT5	IO
GPIO1_IO31	-	EPDC_GDCLK	ALT5	IO

**Table 26-2. GPIO2 External Signals**

Signal	Description	Pad	Mode	Direction
GPIO2_IO00	-	EPDC_GDOE	ALT5	IO
GPIO2_IO01	-	EPDC_GDRL	ALT5	IO
GPIO2_IO02	-	EPDC_GDSP	ALT5	IO
GPIO2_IO03	-	EPDC_VCOM0	ALT5	IO
GPIO2_IO04	-	EPDC_VCOM1	ALT5	IO
GPIO2_IO05	-	EPDC_BDR0	ALT5	IO
GPIO2_IO06	-	EPDC_BDR1	ALT5	IO
GPIO2_IO07	-	EPDC_PWRCTRL0	ALT5	IO
GPIO2_IO08	-	EPDC_PWRCTRL1	ALT5	IO
GPIO2_IO09	-	EPDC_PWRCTRL2	ALT5	IO
GPIO2_IO10	-	EPDC_PWRCTRL3	ALT5	IO
GPIO2_IO11	-	EPDC_PWRCOM	ALT5	IO
GPIO2_IO12	-	EPDC_PWRINT	ALT5	IO
GPIO2_IO13	-	EPDC_PWRSTAT	ALT5	IO
GPIO2_IO14	-	EPDC_PWRWAKEUP	ALT5	IO
GPIO2_IO15	-	LCD_CLK	ALT5	IO
GPIO2_IO16	-	LCD_ENABLE	ALT5	IO
GPIO2_IO17	-	LCD_HSYNC	ALT5	IO
GPIO2_IO18	-	LCD_VSYNC	ALT5	IO
GPIO2_IO19	-	LCD_RESET	ALT5	IO
GPIO2_IO20	-	LCD_DAT0	ALT5	IO
GPIO2_IO21	-	LCD_DAT1	ALT5	IO
GPIO2_IO22	-	LCD_DAT2	ALT5	IO
GPIO2_IO23	-	LCD_DAT3	ALT5	IO
GPIO2_IO24	-	LCD_DAT4	ALT5	IO
GPIO2_IO25	-	LCD_DAT5	ALT5	IO
GPIO2_IO26	-	LCD_DAT6	ALT5	IO
GPIO2_IO27	-	LCD_DAT7	ALT5	IO
GPIO2_IO28	-	LCD_DAT8	ALT5	IO
GPIO2_IO29	-	LCD_DAT9	ALT5	IO
GPIO2_IO30	-	LCD_DAT10	ALT5	IO
GPIO2_IO31	-	LCD_DAT11	ALT5	IO

**Table 26-3. GPIO3 External Signals**

Signal	Description	Pad	Mode	Direction
GPIO3_IO00	-	LCD_DAT12	ALT5	IO
GPIO3_IO01	-	LCD_DAT13	ALT5	IO
GPIO3_IO02	-	LCD_DAT14	ALT5	IO

*Table continues on the next page...*



**Table 26-3. GPIO3 External Signals (continued)**

Signal	Description	Pad	Mode	Direction
GPIO3_IO03	-	LCD_DAT15	ALT5	IO
GPIO3_IO04	-	LCD_DAT16	ALT5	IO
GPIO3_IO05	-	LCD_DAT17	ALT5	IO
GPIO3_IO06	-	LCD_DAT18	ALT5	IO
GPIO3_IO07	-	LCD_DAT19	ALT5	IO
GPIO3_IO08	-	LCD_DAT20	ALT5	IO
GPIO3_IO09	-	LCD_DAT21	ALT5	IO
GPIO3_IO10	-	LCD_DAT22	ALT5	IO
GPIO3_IO11	-	LCD_DAT23	ALT5	IO
GPIO3_IO12	-	I2C1_SCL	ALT5	IO
GPIO3_IO13	-	I2C1_SDA	ALT5	IO
GPIO3_IO14	-	I2C2_SCL	ALT5	IO
GPIO3_IO15	-	I2C2_SDA	ALT5	IO
GPIO3_IO16	-	UART1_RXD	ALT5	IO
GPIO3_IO17	-	UART1_TXD	ALT5	IO
GPIO3_IO18	-	WDOG_B	ALT5	IO
GPIO3_IO19	-	HSIC_DAT	ALT5	IO
GPIO3_IO20	-	HSIC_STROBE	ALT5	IO
GPIO3_IO21	-	REF_CLK_24M	ALT5	IO
GPIO3_IO22	-	REF_CLK_32K	ALT5	IO
GPIO3_IO23	-	PWM1	ALT5	IO
GPIO3_IO24	-	KEY_COL0	ALT5	IO
GPIO3_IO25	-	KEY_ROW0	ALT5	IO
GPIO3_IO26	-	KEY_COL1	ALT5	IO
GPIO3_IO27	-	KEY_ROW1	ALT5	IO
GPIO3_IO28	-	KEY_COL2	ALT5	IO
GPIO3_IO29	-	KEY_ROW2	ALT5	IO
GPIO3_IO30	-	KEY_COL3	ALT5	IO
GPIO3_IO31	-	KEY_ROW3	ALT5	IO

**Table 26-4. GPIO4 External Signals**

Signal	Description	Pad	Mode	Direction
GPIO4_IO00	-	KEY_COL4	ALT5	IO
GPIO4_IO01	-	KEY_ROW4	ALT5	IO
GPIO4_IO02	-	KEY_COL5	ALT5	IO
GPIO4_IO03	-	KEY_ROW5	ALT5	IO
GPIO4_IO04	-	KEY_COL6	ALT5	IO
GPIO4_IO05	-	KEY_ROW6	ALT5	IO

*Table continues on the next page...*

**Table 26-4. GPIO4 External Signals (continued)**

Signal	Description	Pad	Mode	Direction
GPIO4_IO06	-	KEY_COL7	ALT5	IO
GPIO4_IO07	-	KEY_ROW7	ALT5	IO
GPIO4_IO08	-	ECSPI1_SCLK	ALT5	IO
GPIO4_IO09	-	ECSPI1_MOSI	ALT5	IO
GPIO4_IO10	-	ECSPI1_MISO	ALT5	IO
GPIO4_IO11	-	ECSPI1_SS0	ALT5	IO
GPIO4_IO12	-	ECSPI2_SCLK	ALT5	IO
GPIO4_IO13	-	ECSPI2_MOSI	ALT5	IO
GPIO4_IO14	-	ECSPI2_MISO	ALT5	IO
GPIO4_IO15	-	ECSPI2_SS0	ALT5	IO
GPIO4_IO16	-	FEC_TXD1	ALT5	IO
GPIO4_IO17	-	FEC_RXD0	ALT5	IO
GPIO4_IO18	-	FEC_RXD1	ALT5	IO
GPIO4_IO19	-	FEC_RX_ER	ALT5	IO
GPIO4_IO20	-	FEC_MDIO	ALT5	IO
GPIO4_IO21	-	FEC_TX_CLK	ALT5	IO
GPIO4_IO22	-	FEC_TX_EN	ALT5	IO
GPIO4_IO23	-	FEC_MDC	ALT5	IO
GPIO4_IO24	-	FEC_TXD0	ALT5	IO
GPIO4_IO25	-	FEC_CRS_DV	ALT5	IO
GPIO4_IO26	-	FEC_REF_CLK	ALT5	IO
GPIO4_IO27	-	SD2_RST	ALT5	IO
GPIO4_IO28	-	SD2_DAT3	ALT5	IO
GPIO4_IO29	-	SD2_DAT6	ALT5	IO
GPIO4_IO30	-	SD2_DAT1	ALT5	IO
GPIO4_IO31	-	SD2_DAT5	ALT5	IO

**Table 26-5. GPIO5 External Signals**

Signal	Description	Pad	Mode	Direction
GPIO5_IO00	-	SD2_DAT7	ALT5	IO
GPIO5_IO01	-	SD2_DAT0	ALT5	IO
GPIO5_IO02	-	SD2_DAT4	ALT5	IO
GPIO5_IO03	-	SD2_DAT2	ALT5	IO
GPIO5_IO04	-	SD2_CMD	ALT5	IO
GPIO5_IO05	-	SD2_CLK	ALT5	IO
GPIO5_IO06	-	SD1_DAT3	ALT5	IO
GPIO5_IO07	-	SD1_DAT6	ALT5	IO
GPIO5_IO08	-	SD1_DAT1	ALT5	IO

*Table continues on the next page...*

**Table 26-5. GPIO5 External Signals (continued)**

Signal	Description	Pad	Mode	Direction
GPIO5_IO09	-	SD1_DAT5	ALT5	IO
GPIO5_IO10	-	SD1_DAT7	ALT5	IO
GPIO5_IO11	-	SD1_DAT0	ALT5	IO
GPIO5_IO12	-	SD1_DAT4	ALT5	IO
GPIO5_IO13	-	SD1_DAT2	ALT5	IO
GPIO5_IO14	-	SD1_CMD	ALT5	IO
GPIO5_IO15	-	SD1_CLK	ALT5	IO
GPIO5_IO16	-	SD3_DAT2	ALT5	IO
GPIO5_IO17	-	SD3_DAT3	ALT5	IO
GPIO5_IO18	-	SD3_CLK	ALT5	IO
GPIO5_IO19	-	SD3_DAT0	ALT5	IO
GPIO5_IO20	-	SD3_DAT1	ALT5	IO
GPIO5_IO21	-	SD3_CMD	ALT5	IO

## 26.3 Clocks

The table found here describes the clock sources for GPIO.

Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 26-6. GPIO Clocks**

Clock name	Clock Root	Description
ipg_clk_s	ipg_clk_root	Peripheral access clock

## 26.4 GPIO Functional Description

This section provides a complete functional description of the block.

### 26.4.1 GPIO Function

A GPIO signal can operate as a general-purpose input/output when the IOMUX is set to GPIO mode. Each GPIO signal may be independently configured as either an input or an output using the GPIO direction register (GPIO\_GDIR).

When configured as an output (GPIO\_GDIR bit = 1), the value in the data bit in the GPIO data register (GPIO\_DR) is driven on the corresponding GPIO line. When a signal is configured as an input (GPIO\_GDIR bit = 0), the state of the input can be read from the corresponding GPIO\_PSR bit.

## 26.4.2 GPIO Programming

### 26.4.2.1 GPIO Read Mode

The programming sequence for reading input signals should be as follows:

1. Configure IOMUX to select GPIO mode (Via IOMUX Controller (IOMUXC) ).
2. Configure GPIO direction register to input (GPIO\_GDIR[GDIR] set to 0b).
3. Read value from data register/pad status register.

A pseudocode description to read [input3:input0] values is as follows:

```
// SET INPUTS TO GPIO MODE.
write sw_mux_ctl_<input0>_<input1>_<input2>_<input3>, 32'h00000000
// SET GDIR TO INPUT.
write GDIR[31:4,input3_bit, input2_bit, input1_bit, input0_bit,] 32'hxxxxxxx0
// READ INPUT VALUE FROM DR.
read DR
// READ INPUT VALUE FROM PSR.
read PSR
```

#### NOTE

While the GPIO direction is set to input (GPIO\_GDIR = 0), a read access to GPIO\_DR does not return GPIO\_DR data. Instead, it returns the GPIO\_PSR data, which is the corresponding input signal value.

### 26.4.2.2 GPIO Write Mode

The programming sequence for driving output signals should be as follows:

1. Configure IOMUX to select GPIO mode (Via IOMUXC), also enable SION if need to read loopback pad value through PSR
2. Configure GPIO direction register to output (GPIO\_GDIR[GDIR] set to 1b).
3. Write value to data register (GPIO\_DR).

A pseudocode description to drive 4'b0101 on [output3:output0] is as follows:

```
// SET PADS TO GPIO MODE VIA IOMUX.
```

```

write sw_mux_ctl_pad<output[0-3]>.mux_mode, <GPIO_MUX_MODE>
// Enable loopback so we can capture pad value into PSR in output mode
write sw_mux_ctl_pad<output[0-3]>.sion, 1
// SET GDIR=1 TO OUTPUT BITS.
write GDIR[31:4,output3_bit,output2_bit, output1_bit, output0_bit,] 32'hxxxxxxxxF
// WRITE OUTPUT VALUE=4'b0101 TO DR.
write DR, 32'hxxxxxxx5
// READ OUTPUT VALUE FROM PSR ONLY.
read_cmp PSR, 32'hxxxxxxx5

```

### 26.4.3 Interrupt Control Unit

In addition to the general-purpose input/output function, the edge-detect logic in the GPIO peripheral reflects whether a transition has occurred on a given GPIO signal that is configured as an input (GDIR bit = 0). The interrupt control registers (GPIO\_ICR1 and GPIO\_ICR2) may be used to independently configure the interrupt condition of each input signal (low-to-high transition, high-to-low transition, low, or high). For information about GPIO\_ICR1 and GPIO\_ICR2 settings, see [GPIO Memory Map/Register Definition](#).

The interrupt control unit is built of 32 interrupt control subunits, where each subunit handles a single interrupt line.

## 26.5 GPIO Memory Map/Register Definition

There are eight 32-bit GPIO registers. All registers are accessible from the IP interface. Only 32-bit access is supported.

The GPIO memory map is shown in the following table.

**GPIO memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
209_C000	GPIO data register (GPIO1_DR)	32	R/W	0000_0000h	<a href="#">26.5.1/1171</a>
209_C004	GPIO direction register (GPIO1_GDIR)	32	R/W	0000_0000h	<a href="#">26.5.2/1172</a>
209_C008	GPIO pad status register (GPIO1_PSR)	32	R	0000_0000h	<a href="#">26.5.3/1172</a>
209_C00C	GPIO interrupt configuration register1 (GPIO1_ICR1)	32	R/W	0000_0000h	<a href="#">26.5.4/1173</a>
209_C010	GPIO interrupt configuration register2 (GPIO1_ICR2)	32	R/W	0000_0000h	<a href="#">26.5.5/1177</a>
209_C014	GPIO interrupt mask register (GPIO1_IMR)	32	R/W	0000_0000h	<a href="#">26.5.6/1180</a>
209_C018	GPIO interrupt status register (GPIO1_ISR)	32	w1c	0000_0000h	<a href="#">26.5.7/1181</a>
209_C01C	GPIO edge select register (GPIO1_EDGE_SEL)	32	R/W	0000_0000h	<a href="#">26.5.8/1182</a>

## GPIO memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20A_0000	GPIO data register (GPIO2_DR)	32	R/W	0000_0000h	<a href="#">26.5.1/1171</a>
20A_0004	GPIO direction register (GPIO2_GDIR)	32	R/W	0000_0000h	<a href="#">26.5.2/1172</a>
20A_0008	GPIO pad status register (GPIO2_PSR)	32	R	0000_0000h	<a href="#">26.5.3/1172</a>
20A_000C	GPIO interrupt configuration register1 (GPIO2_ICR1)	32	R/W	0000_0000h	<a href="#">26.5.4/1173</a>
20A_0010	GPIO interrupt configuration register2 (GPIO2_ICR2)	32	R/W	0000_0000h	<a href="#">26.5.5/1177</a>
20A_0014	GPIO interrupt mask register (GPIO2_IMR)	32	R/W	0000_0000h	<a href="#">26.5.6/1180</a>
20A_0018	GPIO interrupt status register (GPIO2_ISR)	32	w1c	0000_0000h	<a href="#">26.5.7/1181</a>
20A_001C	GPIO edge select register (GPIO2_EDGE_SEL)	32	R/W	0000_0000h	<a href="#">26.5.8/1182</a>
20A_4000	GPIO data register (GPIO3_DR)	32	R/W	0000_0000h	<a href="#">26.5.1/1171</a>
20A_4004	GPIO direction register (GPIO3_GDIR)	32	R/W	0000_0000h	<a href="#">26.5.2/1172</a>
20A_4008	GPIO pad status register (GPIO3_PSR)	32	R	0000_0000h	<a href="#">26.5.3/1172</a>
20A_400C	GPIO interrupt configuration register1 (GPIO3_ICR1)	32	R/W	0000_0000h	<a href="#">26.5.4/1173</a>
20A_4010	GPIO interrupt configuration register2 (GPIO3_ICR2)	32	R/W	0000_0000h	<a href="#">26.5.5/1177</a>
20A_4014	GPIO interrupt mask register (GPIO3_IMR)	32	R/W	0000_0000h	<a href="#">26.5.6/1180</a>
20A_4018	GPIO interrupt status register (GPIO3_ISR)	32	w1c	0000_0000h	<a href="#">26.5.7/1181</a>
20A_401C	GPIO edge select register (GPIO3_EDGE_SEL)	32	R/W	0000_0000h	<a href="#">26.5.8/1182</a>
20A_8000	GPIO data register (GPIO4_DR)	32	R/W	0000_0000h	<a href="#">26.5.1/1171</a>
20A_8004	GPIO direction register (GPIO4_GDIR)	32	R/W	0000_0000h	<a href="#">26.5.2/1172</a>
20A_8008	GPIO pad status register (GPIO4_PSR)	32	R	0000_0000h	<a href="#">26.5.3/1172</a>
20A_800C	GPIO interrupt configuration register1 (GPIO4_ICR1)	32	R/W	0000_0000h	<a href="#">26.5.4/1173</a>
20A_8010	GPIO interrupt configuration register2 (GPIO4_ICR2)	32	R/W	0000_0000h	<a href="#">26.5.5/1177</a>
20A_8014	GPIO interrupt mask register (GPIO4_IMR)	32	R/W	0000_0000h	<a href="#">26.5.6/1180</a>
20A_8018	GPIO interrupt status register (GPIO4_ISR)	32	w1c	0000_0000h	<a href="#">26.5.7/1181</a>
20A_801C	GPIO edge select register (GPIO4_EDGE_SEL)	32	R/W	0000_0000h	<a href="#">26.5.8/1182</a>
20A_C000	GPIO data register (GPIO5_DR)	32	R/W	0000_0000h	<a href="#">26.5.1/1171</a>
20A_C004	GPIO direction register (GPIO5_GDIR)	32	R/W	0000_0000h	<a href="#">26.5.2/1172</a>
20A_C008	GPIO pad status register (GPIO5_PSR)	32	R	0000_0000h	<a href="#">26.5.3/1172</a>
20A_C00C	GPIO interrupt configuration register1 (GPIO5_ICR1)	32	R/W	0000_0000h	<a href="#">26.5.4/1173</a>
20A_C010	GPIO interrupt configuration register2 (GPIO5_ICR2)	32	R/W	0000_0000h	<a href="#">26.5.5/1177</a>
20A_C014	GPIO interrupt mask register (GPIO5_IMR)	32	R/W	0000_0000h	<a href="#">26.5.6/1180</a>
20A_C018	GPIO interrupt status register (GPIO5_ISR)	32	w1c	0000_0000h	<a href="#">26.5.7/1181</a>
20A_C01C	GPIO edge select register (GPIO5_EDGE_SEL)	32	R/W	0000_0000h	<a href="#">26.5.8/1182</a>

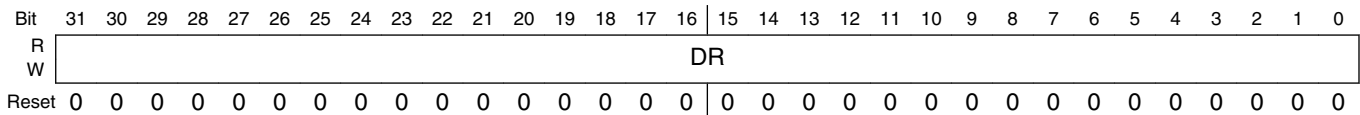
## 26.5.2 GPIO data register (GPIOx\_DR)

The 32-bit GPIO\_DR register stores data that is ready to be driven to the output lines. If the IOMUXC is in GPIO mode and a given GPIO direction bit is set, then the corresponding DR bit is driven to the output. If a given GPIO direction bit is cleared, then a read of GPIO\_DR reflects the value of the corresponding signal. Two wait states are required in read access for synchronization.

The results of a read of a DR bit depends on the IOMUXC input mode settings and the corresponding GDIR bit as follows:

- If GDIR[n] is set and IOMUXC input mode is GPIO, then reading DR[n] returns the contents of DR[n].
- If GDIR[n] is cleared and IOMUXC input mode is GPIO, then reading DR[n] returns the corresponding input signal's value.
- If GDIR[n] is set and IOMUXC input mode is not GPIO, then reading DR[n] returns the contents of DR[n].
- If GDIR[n] is cleared and IOMUXC input mode is not GPIO, then reading DR[n] always returns zero.

Address: Base address + 0h offset



### GPIOx\_DR field descriptions

Field	Description
31–0 DR	<p>Data bits. This register defines the value of the GPIO output when the signal is configured as an output (GDIR[n]=1). Writes to this register are stored in a register. Reading GPIO_DR returns the value stored in the register if the signal is configured as an output (GDIR[n]=1), or the input signal's value if configured as an input (GDIR[n]=0).</p> <p><b>NOTE:</b> The I/O multiplexer must be configured to GPIO mode for the GPIO_DR value to connect with the signal. Reading the data register with the input path disabled always returns a zero value.</p>

### 26.5.3 GPIO direction register (GPIOx\_GDIR)

GPIO\_GDIR functions as direction control when the IOMUXC is in GPIO mode. Each bit specifies the direction of a one-bit signal. The mapping of each DIR bit to a corresponding SoC signal is determined by the SoC's pin assignment and the IOMUX table. For more details consult the IOMUXC chapter.

Address: Base address + 4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GDIR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### GPIOx\_GDIR field descriptions

Field	Description
31–0 GDIR	GPIO direction bits. Bit n of this register defines the direction of the GPIO[n] signal.  <b>NOTE:</b> GPIO_GDIR affects only the direction of the I/O signal when the corresponding bit in the I/O MUX is configured for GPIO.  0 <b>INPUT</b> — GPIO is configured as input. 1 <b>OUTPUT</b> — GPIO is configured as output.

### 26.5.4 GPIO pad status register (GPIOx\_PSR)

GPIO\_PSR is a read-only register. Each bit stores the value of the corresponding input signal (as configured in the IOMUX). This register is clocked with the ipg\_clk\_s clock, meaning that the input signal is sampled only when accessing this location. Two wait states are required any time this register is accessed for synchronization.

Address: Base address + 8h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PSR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### GPIOx\_PSR field descriptions

Field	Description
31–0 PSR	GPIO pad status bits (status bits). Reading GPIO_PSR returns the state of the corresponding input signal.



**GPIOx\_PSR field descriptions (continued)**

Field	Description
	Settings:  <b>NOTE:</b> The IOMUXC must be configured to GPIO mode for GPIO_PSR to reflect the state of the corresponding signal.

**26.5.5 GPIO interrupt configuration register1 (GPIOx\_ICR1)**

GPIO\_ICR1 contains 16 two-bit fields, where each field specifies the interrupt configuration for a different input signal.

Address: Base address + Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
	ICR15	ICR14	ICR13	ICR12	ICR11	ICR10	ICR9	ICR8								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
	ICR7	ICR6	ICR5	ICR4	ICR3	ICR2	ICR1	ICR0								
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**GPIOx\_ICR1 field descriptions**

Field	Description
31–30 ICR15	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 15.  Settings:  Bits ICRn[1:0] determine the interrupt condition for signal n as follows:  00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive. 01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive. 10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive. 11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.
29–28 ICR14	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 14.  Settings:  Bits ICRn[1:0] determine the interrupt condition for signal n as follows:  00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive. 01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive. 10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive. 11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.
27–26 ICR13	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 13.

*Table continues on the next page...*

## GPIOx\_ICR1 field descriptions (continued)

Field	Description
	<p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
25–24 ICR12	<p>Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 12.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
23–22 ICR11	<p>Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 11.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
21–20 ICR10	<p>Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 10.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
19–18 ICR9	<p>Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 9.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
17–16 ICR8	<p>Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 8.</p> <p>Settings:</p>

*Table continues on the next page...*

**GPIOx\_ICR1 field descriptions (continued)**

Field	Description
	<p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
15–14 ICR7	<p>Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 7.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
13–12 ICR6	<p>Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 6.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
11–10 ICR5	<p>Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 5.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
9–8 ICR4	<p>Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 4.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
7–6 ICR3	<p>Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 3.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p>

*Table continues on the next page...*

## GPIOx\_ICR1 field descriptions (continued)

Field	Description
	00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive. 01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive. 10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive. 11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.
5–4 ICR2	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 2.  Settings:  Bits ICRn[1:0] determine the interrupt condition for signal n as follows:  00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive. 01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive. 10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive. 11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.
3–2 ICR1	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 1.  Settings:  Bits ICRn[1:0] determine the interrupt condition for signal n as follows:  00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive. 01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive. 10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive. 11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.
1–0 ICR0	Interrupt configuration 1 fields. This register controls the active condition of the interrupt function for GPIO interrupt 0.  Settings:  Bits ICRn[1:0] determine the interrupt condition for signal n as follows:  00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive. 01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive. 10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive. 11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.

## 26.5.6 GPIO interrupt configuration register2 (GPIOx\_ICR2)

GPIO\_ICR2 contains 16 two-bit fields, where each field specifies the interrupt configuration for a different input signal.

Address: Base address + 10h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
	ICR31		ICR30		ICR29		ICR28		ICR27		ICR26		ICR25		ICR24	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
	ICR23		ICR22		ICR21		ICR20		ICR19		ICR18		ICR17		ICR16	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### GPIOx\_ICR2 field descriptions

Field	Description
31–30 ICR31	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 31.  Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows:  00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive. 01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive. 10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive. 11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.
29–28 ICR30	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 30.  Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows:  00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive. 01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive. 10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive. 11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.
27–26 ICR29	Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 29.  Settings: Bits ICRn[1:0] determine the interrupt condition for signal n as follows:  00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive. 01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive. 10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive. 11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.

*Table continues on the next page...*

**GPIOx\_ICR2 field descriptions (continued)**

Field	Description
25–24 ICR28	<p>Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 28.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
23–22 ICR27	<p>Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 27.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
21–20 ICR26	<p>Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 26.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
19–18 ICR25	<p>Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 25.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
17–16 ICR24	<p>Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 24.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>

*Table continues on the next page...*

**GPIOx\_ICR2 field descriptions (continued)**

Field	Description
15–14 ICR23	<p>Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 23.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.  01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.  10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.  11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
13–12 ICR22	<p>Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 22.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.  01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.  10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.  11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
11–10 ICR21	<p>Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 21.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.  01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.  10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.  11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
9–8 ICR20	<p>Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 20.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.  01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.  10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.  11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
7–6 ICR19	<p>Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 19.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.  01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.  10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.  11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>

*Table continues on the next page...*

**GPIOx\_ICR2 field descriptions (continued)**

Field	Description
5–4 ICR18	<p>Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 18.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
3–2 ICR17	<p>Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 17.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>
1–0 ICR16	<p>Interrupt configuration 2 fields. This register controls the active condition of the interrupt function for GPIO interrupt 16.</p> <p>Settings:</p> <p>Bits ICRn[1:0] determine the interrupt condition for signal n as follows:</p> <p>00 <b>LOW_LEVEL</b> — Interrupt n is low-level sensitive.</p> <p>01 <b>HIGH_LEVEL</b> — Interrupt n is high-level sensitive.</p> <p>10 <b>RISING_EDGE</b> — Interrupt n is rising-edge sensitive.</p> <p>11 <b>FALLING_EDGE</b> — Interrupt n is falling-edge sensitive.</p>

**26.5.7 GPIO interrupt mask register (GPIOx\_IMR)**

GPIO\_IMR contains masking bits for each interrupt line.

Address: Base address + 14h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	<div>IMR</div>																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**GPIOx\_IMR field descriptions**

Field	Description
31–0 IMR	Interrupt Mask bits. This register is used to enable or disable the interrupt function on each of the 32 GPIO signals.



**GPIOx\_IMR field descriptions (continued)**

Field	Description
	Settings: Bit IMR[n] (n=0...31) controls interrupt n as follows:  0 <b>UNMASKED</b> — Interrupt n is disabled. 1 <b>MASKED</b> — Interrupt n is enabled.

**26.5.8 GPIO interrupt status register (GPIOx\_ISR)**

The GPIO\_ISR functions as an interrupt status indicator. Each bit indicates whether an interrupt condition has been met for the corresponding input signal. When an interrupt condition is met (as determined by the corresponding interrupt condition register field), the corresponding bit in this register is set. Two wait states are required in read access for synchronization. One wait state is required for reset.

Address: Base address + 18h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																	ISR															
W																	w1c															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

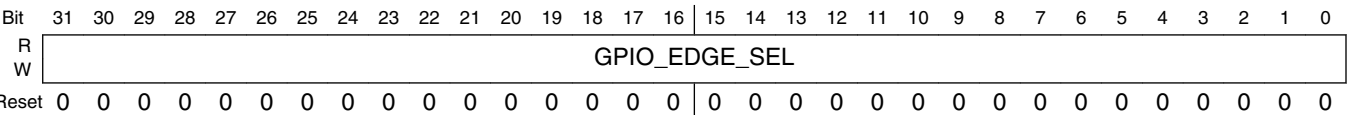
**GPIOx\_ISR field descriptions**

Field	Description
31–0 ISR	Interrupt status bits - Bit n of this register is asserted (active high) when the active condition (as determined by the corresponding ICR bit) is detected on the GPIO input and is waiting for service. The value of this register is independent of the value in GPIO_IMR.  When the active condition has been detected, the corresponding bit remains set until cleared by software. Status flags are cleared by writing a 1 to the corresponding bit position.

### 26.5.9 GPIO edge select register (GPIOx\_EDGE\_SEL)

GPIO\_EDGE\_SEL may be used to override the ICR registers' configuration. If the GPIO\_EDGE\_SEL bit is set, then a rising edge or falling edge in the corresponding signal generates an interrupt. This register provides backward compatibility. On reset all bits are cleared (ICR is not overridden).

Address: Base address + 1Ch offset



GPIOx\_EDGE\_SEL field descriptions

Field	Description
31–0 GPIO_EDGE_SEL	Edge select. When GPIO_EDGE_SEL[n] is set, the GPIO disregards the ICR[n] setting, and detects any edge on the corresponding input signal.

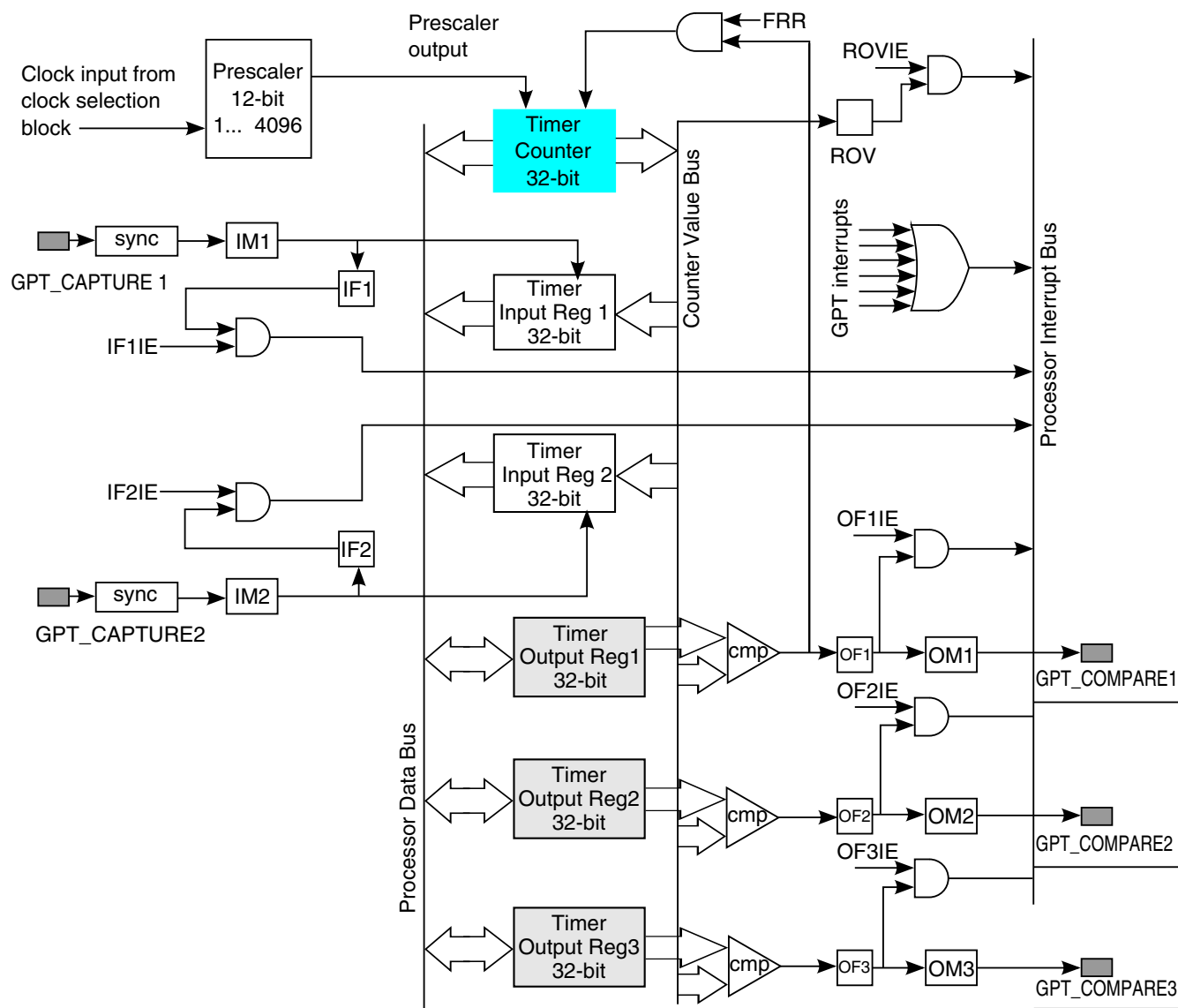
# Chapter 27

## General Purpose Timer (GPT)

### 27.1 Overview

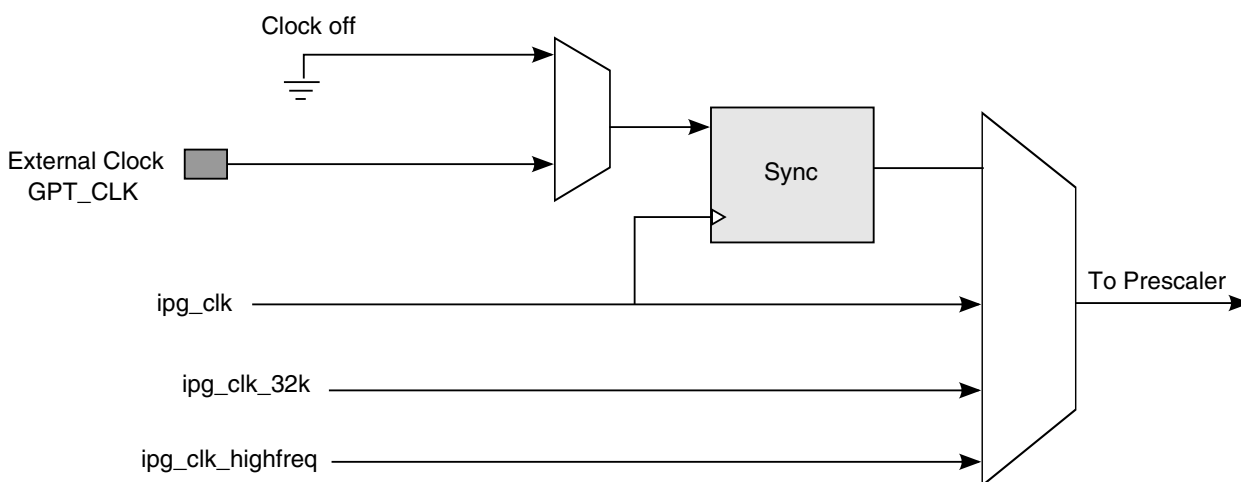
This chapter describes the General Purpose Timer (GPT) module interface. It is also a reference for software driver programming.

The GPT has a 32-bit up-counter. The timer counter value can be captured in a register using an event on an external pin. The capture trigger can be programmed to be a rising or/and falling edge. The GPT can also generate an event on the DO\_CMPOUT $n$  pins and an interrupt when the timer reaches a programmed value. The GPT has a 12-bit prescaler, which provides a programmable clock frequency derived from multiple clock sources.



**Figure 27-1. GPT Block Diagram**

The following figure shows the GPT functional clocking scheme.



**Figure 27-2. GPT Counter Clocks Diagram**

### 27.1.1 Features

- One 32-bit up-counter with clock source selection, including external clock.
- Two input capture channels with a programmable trigger edge.
- Three output compare channels with a programmable output mode. A "forced compare" feature is also available.
- Can be programmed to be *active* in low power and debug modes.
- Interrupt generation at capture, compare, and rollover events.
- Restart or free-run modes for counter operations.

### 27.1.2 Modes and Operation

The GPT supports the modes described in the indicated sections:

- [Operating Modes](#)
  - [Restart Mode](#)
  - [Free-Run Mode](#)

## 27.2 External Signals

The GPT follows the IP Bus protocol for interfacing with the processor core. The GPT does not have *any interface signals with any other module inside the chip*, except for the clock and reset inputs (from the clock and reset controller module) and for the interrupt signals *to* the processor interrupt handler. There are functional and clock inputs, and functional output signals going outside the chip boundary.

The following table describes all block signals that connect off-chip.

**Table 27-1. GPT External Signals**

Signal	Description	Pad	Mode	Direction
GPT_CLK	Input pin for an external clock that the counter can be operated at.	FEC_TXD0	ALT4	I
		LCD_DAT23	ALT4	
GPT_CAPTURE1	Input pin for a capture event for Input Capture Channel 1.	FEC_MDIO	ALT4	I
		LCD_DAT18	ALT4	
GPT_CAPTURE2	Input pin for a capture event for Input Capture Channel 2.	FEC_TX_CLK	ALT4	I
		LCD_DAT19	ALT4	
GPT_COMPARE1	Output pin that indicates a "compare event" occurrence in Output Compare Channel 1.	FEC_RX_ER	ALT4	O
		LCD_DAT20	ALT4	
GPT_COMPARE2	Output pin that indicates a "compare event" occurrence in Output Compare Channel 2.	FEC CRS_DV	ALT4	O
		LCD_DAT21	ALT4	
GPT_COMPARE3	Output pin that indicates a "compare event" occurrence in Output Compare Channel 3.	FEC_RXD1	ALT4	O
		LCD_DAT22	ALT4	

There are six signals (three input, three output) in the GPT module that *can be* connected to the chip pads.

### 27.2.1 External Clock Input

The GPT counter can be operated using an external clock from outside the device, and this is the input pin used for that purpose.

The external clock input GPT\_CLK is treated as asynchronous to the peripheral clock. To ensure proper operations of GPT, the external clock input frequency should be less than 1/4 of frequency of the peripheral clock. Hysteresis characteristics on this pad will be required because this is a clock input.

## 27.2.2 Input Capture Trigger Signals

The GPT counter value can be stored in a register, triggered by an event from *outside the device*.

A positive or/and negative edge on these signals GPT\_CAPTURE1 , GPT\_CAPTURE2 can trigger this capture event. These signals are treated as asynchronous to the peripheral clock. Only those transitions which occur *at least a single clock cycle* (the clock selected to run the counter) *after the previous recorded transition* are guaranteed to trigger a capture event.

## 27.2.3 Output Compare Signals

The output compare signals: GPT\_COMPARE1, GPT\_COMPARE2, GPT\_COMPARE3, indicate that output compare events have gone through a specified transition.

## 27.3 Clocks

The clock that is input to the prescaler can be selected from 4 clock sources:

The following table describes the clock sources for GPT. Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 27-2. GPT Clocks**

Clock name	Clock Root	Description
ipg_clk	ipg_clk_root	Peripheral clock
ipg_clk_32k	ckil_sync_clk_root	Low-frequency reference clock (32kHz)
ipg_clk_highfreq	perclk_clk_root	High-frequency reference clock
ipg_clk_s	ipg_clk_root	Peripheral access clock

- High-Frequency Clock (ipg\_clk\_highfreq)

Provided by the Clock Controller Module (CCM), the High Frequency Clock is intended to be ON in Normal Power mode when the Peripheral Clock (ipg\_clk) is turned OFF, thereby enabling the GPT to be operated using the High Frequency Clock *in Normal Power mode*. The CCM is expected to provide this clock *after* synchronizing it to the System Bus Clock in Normal functional mode; the CCM is also expected to switch to the *unsynchronized* version of the High Frequency Clock in a Low Power mode.

- Low-Reference Clock (ipg\_clk\_32k)

This 32 kHz Low Reference Clock (provided by the CCM) is intended to be ON in Low Power mode when the Peripheral Clock (ipg\_clk) is turned OFF, thereby enabling the GPT to be operated using the Low Reference Clock in Low Power mode. The CCM is expected to provide the Low Reference Clock *after* synchronizing it to the System Bus Clock in Normal functional mode; the CCM is also expected to switch to the *unsynchronized* version of the Low Reference Clock in a Low Power mode.

- Peripheral Clock (ipg\_clk)

If the Peripheral Clock or the External Clock is selected (CLKSRC=001 or 011) as Clock Source, then the Peripheral Clock will be ON in normal GPT operations. In Low Power modes, if the GPT is programmed to be disabled (STOPEN or WAITEN or DOZEN=0), then the Peripheral Clock can be switched OFF.

- External Clock

The External Clock comes from *outside the device* and can be selected to run the GPT counter. The External Clock is treated as *asynchronous to the Peripheral Clock*, and is synchronized to the Peripheral Clock, *inside* the module. Therefore, the External Clock frequency is limited to  $< 1/4$  frequency of the Peripheral Clock, for proper GPT operations. Note that in Low Power modes, *if* the Peripheral Clock is not available, then the External Clock *cannot be used* to run the counter.

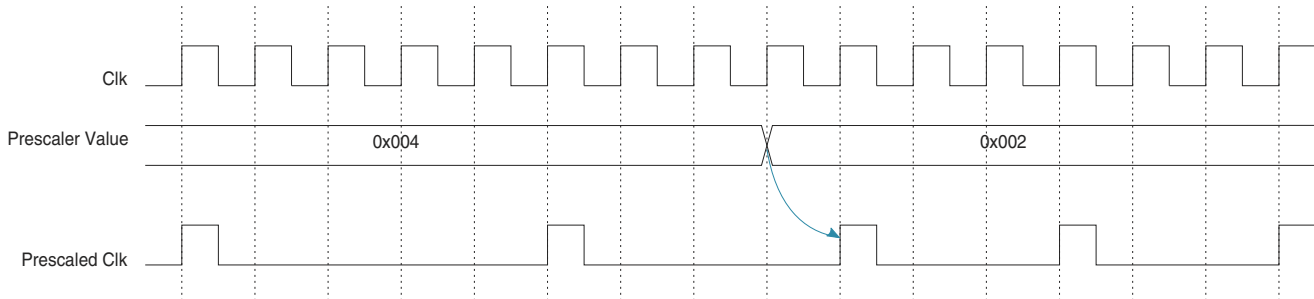
- Crystal Oscillator Clock

This 24MHz Crystal Oscillator Clock (provided by the CCM) is intended to be used against frequency change of Peripheral Clock changes to provide a more accurate timer clock for operation system. The CCM is expected to provide the 24MHz Crystal Oscillator Clock *without* synchronizing it to the System Bus Clock in Normal functional mode. Synchronization is done in GPT module. Before synchronization, the 24MHz Crystal Oscillator Clock is divided by a 24MHz clock prescaler, to make sure the clock frequency less than half of System Bus Clock .

The clock input source is configured using the clock source field (CLKSRC, in the GPT\_CR control register). The clock input to the prescaler can be disabled by programming the CLKSRC bits (of the GPT\_CR control register) to 000. **The CLKSRC field value should be changed only after disabling the GPT** (by setting the EN bit in the GPT\_CR to 0).



The PRESCALER field selects the divide ratio of the input clock that drives the main counter. The prescaler can divide the input clock by a value (from 1 to 4096) and can be changed *at any time*. A change in the value of the PRESCALER field *immediately affects* the output clock frequency.



**Figure 27-3. Prescaler Value Change Timing Diagram**

## 27.4 Functional Description

This section provides a complete functional description of the GPT.

### 27.4.1 Operating Modes

The GPT counter can be programmed to work in either of two modes: Restart mode or Free-Run mode.

#### 27.4.1.1 Restart Mode

In Restart mode (selectable through the GPT Control Register GPT\_CR), when the counter reaches the compared value, the counter resets and starts again from 0x00000000. The Restart feature is associated only with Compare Channel 1.

Any write access to the Compare register of Channel 1 will reset the GPT counter. This is done to avoid possibly missing a compare event when compare value is changed from a higher value to lower value while counting is proceeding.

For the other two compare channels, when the compare event occurs the counter is *not reset*.

### 27.4.1.2 Free-Run Mode

In Free-Run mode, when compare events occur for all 3 channels, the counter is *not reset*; instead the counter continues to count until 0xffffffff, and then rolls over (to 0x00000000).

### 27.4.2 Operation

The General Purpose Timer (GPT) has a single counter (GPT\_CNT) that is a 32-bit free-running *up-counter*, which starts counting *after it is enabled by software* (EN=1).

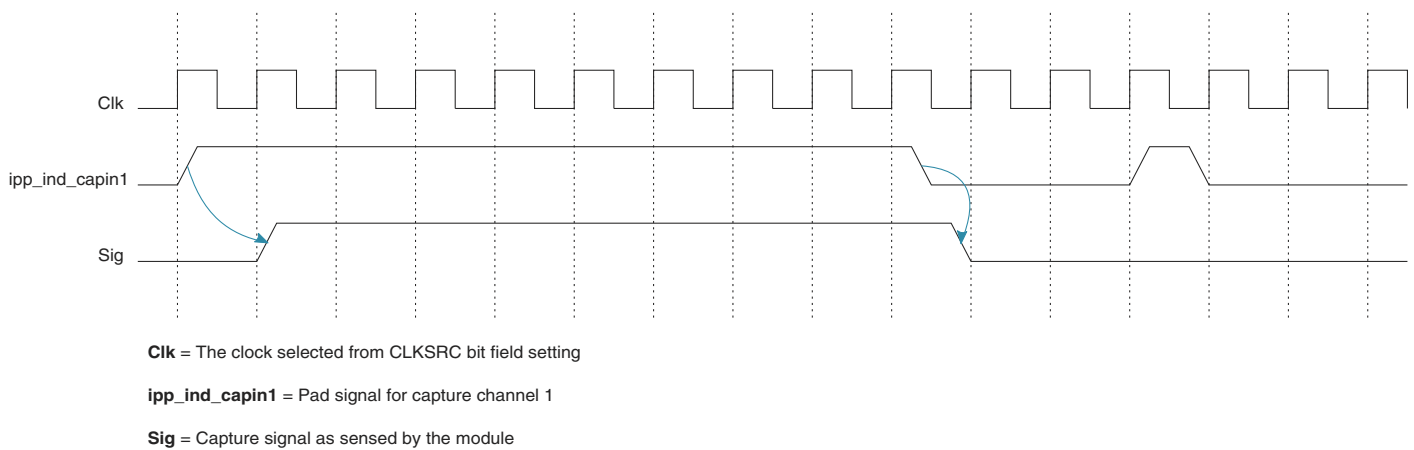
The counter's clock source is the output of the prescaler labelled "Prescaler output" in [Figure 27-1](#).

- If the GPT timer is disabled (EN=0), then the Main Counter *and* Prescaler Counter freeze their current count values. The ENMOD bit determines the value of the GPT counter when the EN bit is set and the Counter is enabled again.
  - If the ENMOD bit is set (=1), then the Main Counter and Prescaler Counter values are reset to 0, when GPT is enabled (EN=1).
  - If ENMOD bit is programmed to 0, then the Main Counter and Prescaler Counter restart counting from their frozen values, when GPT is enabled again (EN=1).
- If GPT is programmed to be disabled in a low power mode (STOP/WAIT), then the Main Counter and Prescaler Counter freeze at their current count values *when* GPT enters low power mode. When GPT exits a low power mode, the Main Counter and Prescaler Counter start counting from their frozen values *regardless* of the ENMOD bit value. Note that the GPT\_CNT can be read *at any time* by the processor, and that *both* Input Capture Channels use the *same* counter (GPT\_CNT).
- A hardware reset resets all the GPT registers to their respective reset values. All registers except the Output Compare Registers (OCR1, OCR2, OCR3) obtain a value of 0x0. The Compare registers are reset to 0xffffffff.
- The software reset (SWR bit in the GPT\_CR control register) resets *all* of the register bits *except* the EN, ENMOD, STOPEN, WAITEN, and DBGGEN bits. The state of these bits is not affected by a software reset. Note that a software reset can be given *while the GPT is disabled*.

#### 27.4.2.1 Input Capture

There are two Input Capture Channels, and each Input Capture Channel has a dedicated capture pin, capture register and input edge detection/selection logic. Each input capture function has an associated status flag, and can cause the processor to make an interrupt service request.

When a selected edge transition occurs on an Input Capture pin, the contents of the GPT\_CNT is captured on the corresponding capture register and the appropriate interrupt status flag is set. An interrupt request can be generated when the transition is detected *if* its corresponding enable bit is set (in the Interrupt Register). The capture can be programmed to occur on the input pin's rising edge, falling edge, on both rising and falling edges, or the capture can be disabled. The events are synchronized with the clock that was selected to run the counter. Only those transitions that occur at least one clock cycle *after* the previous recorded transition will be guaranteed to trigger a capture event. There can be up to one clock cycle of uncertainty in the latching of the input transition. The Input Capture registers can be read *at any time* without affecting their values.

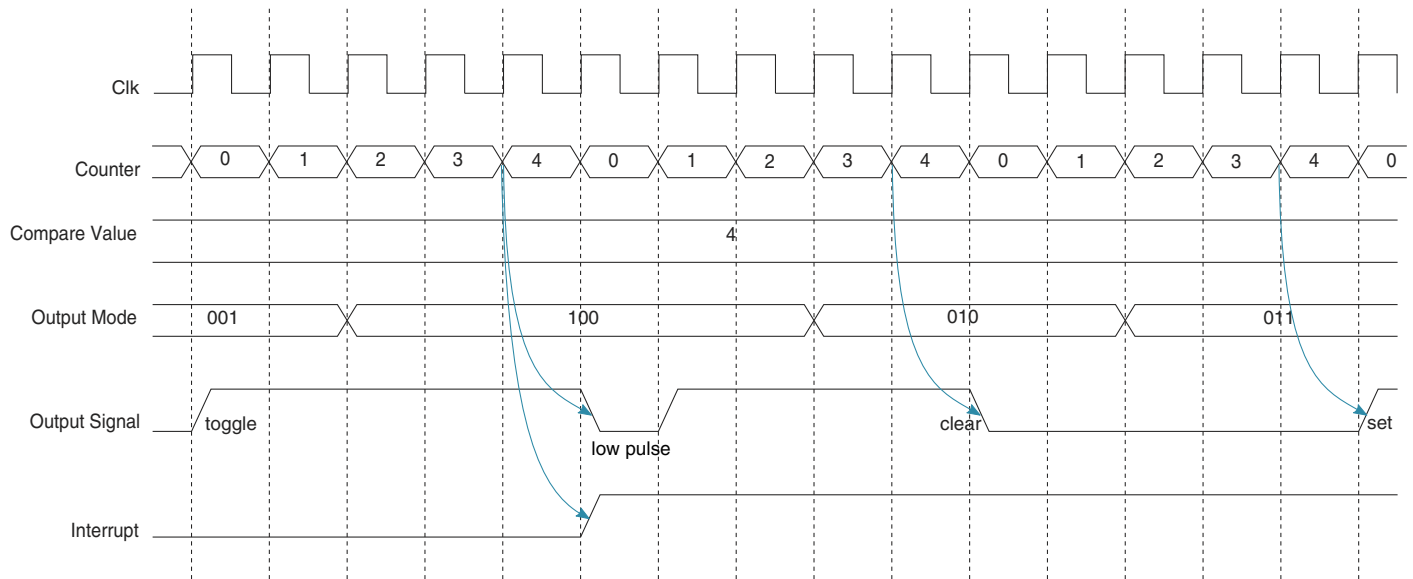


**Figure 27-4. Input Capture Event Timing**

### 27.4.2.2 Output Compare

The three Output Compare Channels *use the same counter* (GPT\_CNT) as the Input Capture Channels. When the programmed content of an Output Compare register matches the value in GPT\_CNT, an output compare status flag is set and an interrupt is generated (if the corresponding bit is set in the interrupt register). Consequently, the Output Compare timer pin will be set, cleared, toggled, not affected at all or provide an active-low pulse for one input clock period (subject to the restriction on the maximum frequency allowed on the pad) according to the mode bits (that were programmed).

There is also a "forced-compare" feature that allows the software to generate a compare event when required, *without the condition of the counter value that is equal to the compare value*. The action taken as a result of a forced compare is the same as when an output compare match occurs, *except that the status flags are not set and no interrupt can be generated*. Forced channels take programmed action immediately after the write to the force-compare bits. These bits are self-negating and always read as zeros.



**Figure 27-5. Output Compare and Interrupt Timing**

### 27.4.2.3 Interrupts

There are 6 different interrupts that are generated by the GPT. If the selected clock for running the counter is available, then *all interrupts can be generated in Low Power and Debug modes*.

- Rollover Interrupt

The Rollover Interrupt is generated when the GPT counter reaches 0xffffffff, then resets to 0x00000000 and continues counting. The Rollover Interrupt is enabled by the ROVIE bit in the GPT\_IR register; the associated status bit is the ROV bit in the GPT\_SR register.

- Input Capture Interrupt 1, 2

After a capture event occurs, the associated Input Capture Channel generates an interrupt. The "capture event" interrupts are enabled by the IF2IE and IF1IE bits (in the GPT\_IR register); the associated status bits are IF2 and IF1 (in the GPT\_SR register). The capture of the counter value because of a capture event is *not affected by a pending capture interrupt*. The Capture register is updated with a new counter value when a capture event occurs, regardless of whether that Capture Channels' interrupt has been serviced or not.

- Output Compare Interrupt 1, 2, 3

After a compare event occurs, the associated Output Compare Channel generates an interrupt. The "compare event" interrupts are enabled by the OF3IE, OF2IE, and OF1IE bits (in the GPT\_IR register); the associated status bits are OF3, OF2, and OF1 (in the GPT\_SR register). A "forced compare" does not generate an interrupt.

A *cumulative* interrupt line is also present, which is asserted whenever any of the above interrupts are posted. The cumulative interrupt line has *no* associated enables or status bits.

#### 27.4.2.4 Low Power Mode Behavior

In Low Power modes, if the clock from the selected clock source is available (except for the External Clock, which can be used *only if* the Peripheral Clock is available), the counter will continue to run depending on whether the control bit for that mode is set. If the clock is not present or if the corresponding low power bit in the GPT\_CR control register is 0, the Main Counter and the Prescaler Counter freeze at their current values and resume counting (from their frozen values) when the Low Power mode is exited.

#### 27.4.2.5 Debug Mode Behavior

In Debug mode, the modules in the device have the option of continuing to run or be halted.

- If the DBGEN bit is set, then the GPT timer will continue to run in Debug mode.
- If the DBGEN bit is not set (in the GPT\_CR control register), then the GPT timer is halted.

### 27.5 Initialization/ Application Information

#### 27.5.1 Selecting the Clock Source

The CLKSRC field in the GPT\_CR register selects the clock source. The CLKSRC field value should be changed only after disabling the GPT (EN=0).

The software sequence to be followed while changing clock source is:

1. Disable GPT by setting EN=0 in GPT\_CR register.
2. Disable GPT interrupt register (GPT\_IR).

3. Configure Output Mode to unconnected/ disconnected—Write zeros in OM3, OM2, and OM1 in GPT\_CR
4. Disable Input Capture Modes—Write zeros in IM1 and IM2 in GPT\_CR
5. Change clock source CLKSRC to the desired value in GPT\_CR register.
6. Assert the SWR bit in GPT\_CR register.
7. Clear GPT status register (GPT\_SR) (i.e., w1c).
8. Set ENMOD=1 in GPT\_CR register, to bring GPT counter to 0x00000000.
9. Enable GPT (EN=1) in GPT\_CR register.
10. Enable GPT interrupt register (GPT\_IR).

## 27.6 GPT Memory Map/Register Definition

The GPT has 10 user-accessible 32-bit registers, which are used to configure, operate, and monitor the state of the GPT.

An IP bus write access to the GPT Control Register (GPT\_CR) and the GPT Output Compare Register1 (GPT\_OCR1) results in *one cycle of wait state*, while other valid IP bus accesses incur 0 wait states.

Irrespective of the Response Select signal value, a Write access to the GPT Status Registers (Read-only registers GPT\_ICR1, GPT\_ICR2, GPT\_CNT) will generate a bus exception.

- If the Response Select signal is driven Low, then the Read/Write access to the *unimplemented* address space of GPT (*ips\_addr* is greater than or equal to \$BASE + \$028) will generate a bus exception.
- If the Response Select is driven High, then the Read/Write access to the unimplemented address space of GPT will *not* generate any error response (like a bus exception).

**GPT memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
209_8000	GPT Control Register (GPT_CR)	32	R/W	0000_0000h	<a href="#">27.6.1/1195</a>
209_8004	GPT Prescaler Register (GPT_PR)	32	R/W	0000_0000h	<a href="#">27.6.2/1199</a>
209_8008	GPT Status Register (GPT_SR)	32	R/W	0000_0000h	<a href="#">27.6.3/1200</a>
209_800C	GPT Interrupt Register (GPT_IR)	32	R/W	0000_0000h	<a href="#">27.6.4/1201</a>
209_8010	GPT Output Compare Register 1 (GPT_OCR1)	32	R/W	FFFF_FFFFh	<a href="#">27.6.5/1202</a>
209_8014	GPT Output Compare Register 2 (GPT_OCR2)	32	R/W	FFFF_FFFFh	<a href="#">27.6.6/1203</a>

*Table continues on the next page...*

**GPT memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
209_8018	GPT Output Compare Register 3 (GPT_OCR3)	32	R/W	FFFF_FFFFh	<a href="#">27.6.7/1203</a>
209_801C	GPT Input Capture Register 1 (GPT_ICR1)	32	R	0000_0000h	<a href="#">27.6.8/1204</a>
209_8020	GPT Input Capture Register 2 (GPT_ICR2)	32	R	0000_0000h	<a href="#">27.6.9/1204</a>
209_8024	GPT Counter Register (GPT_CNT)	32	R	0000_0000h	<a href="#">27.6.10/1205</a>

**27.6.1 GPT Control Register (GPT\_CR)**

The GPT Control Register (GPT\_CR) is used to program and configure GPT operations. An IP Bus Write to the GPT Control Register occurs after one cycle of wait state, while an IP Bus Read occurs after 0 wait states.

Address: 209\_8000h base + 0h offset = 209\_8000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	OM3			OM2			OM1			IM2		IM1	
W	FO3	FO2	FO1													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
R	SWR			0			24MEN		FRR		CLKSRC		STOPEN	DOZEEN	WAITEN	DBGEN	ENMOD	EN
W																		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

**GPT\_CR field descriptions**

Field	Description
31 FO3	FO3 Force Output Compare Channel 3
	FO2 Force Output Compare Channel 2

Table continues on the next page...

## GPT\_CR field descriptions (continued)

Field	Description
	<p>FO1 Force Output Compare Channel 1</p> <p>The FOn bit causes the pin action <i>programmed</i> for the timer Output Compare <i>n</i> pin (according to the OMn bits in this register).</p> <ul style="list-style-type: none"> <li>The OFn flag (OF3, OF2, OF1) in the status register is <b>not affected</b>.</li> <li>This bit is self-negating and always read as zero.</li> </ul> <p>0 Writing a 0 has no effect.</p> <p>1 Causes the programmed pin action on the timer Output Compare <i>n</i> pin; the OFn flag is not set.</p>
30 FO2	See F03
29 FO1	See F03
28–26 OM3	<p>OM3 (bits 28-26) controls the Output Compare Channel 3 operating mode.</p> <p>OM2 (bits 25-23) controls the Output Compare Channel 2 operating mode.</p> <p>OM1 (bits 22-20) controls the Output Compare Channel 1 operating mode.</p> <p>The OMn bits specify the response that a compare event will generate on the output pin of Output Compare Channel <i>n</i>.</p> <ul style="list-style-type: none"> <li>The toggle, clear, and set options cause a change on the output pin <i>only</i> if a compare event occurs.</li> <li>When OMn is programmed as 1xx (active low pulse), the output pin is set to one immediately on the next input clock; a low pulse (that is an input clock in width) occurs when there is a compare event. Note that here, "input clock" refers to the clock selected by the CLKSRC bits of the GPT Control Register.</li> </ul> <p>000 Output disconnected. No response on pin.</p> <p>001 Toggle output pin</p> <p>010 Clear output pin</p> <p>011 Set output pin</p> <p>1xx Generate an active low pulse (that is one input clock wide) on the output pin.</p>
25–23 OM2	See OM3
22–20 OM1	See OM3
19–18 IM2	<p>IM2 (bits 19-18, Input Capture Channel 2 operating mode)</p> <p>IM1 (bits 17-16, Input Capture Channel 1 operating mode)</p> <p>The IMn bit field determines the transition on the input pin (for Input capture channel <i>n</i>), which will trigger a capture event.</p> <p>00 capture disabled</p> <p>01 capture on rising edge only</p> <p>10 capture on falling edge only</p> <p>11 capture on both edges</p>
17–16 IM1	See IM2
15 SWR	<p>Software reset.</p> <p>This is the software reset of the GPT module. It is a self-clearing bit.</p> <ul style="list-style-type: none"> <li>The SWR bit is set when the module is in reset state.</li> </ul>

*Table continues on the next page...*



## GPT\_CR field descriptions (continued)

Field	Description
	<ul style="list-style-type: none"> <li>The SWR bit is cleared when the reset procedure finishes.</li> <li>Setting the SWR bit resets <b>all of the registers</b> to their default reset values, except for the CLKSRC, EN, ENMOD, STOPEN, WAITEN, and DBGEN bits in the GPT Control Register (this control register).</li> </ul> <p>0 GPT is not in reset state 1 GPT is in reset state</p>
14–11 Reserved	This read-only field is reserved and always has the value 0.
10 24MEN	<p>Enable 24MHz clock input from crystal.</p> <ul style="list-style-type: none"> <li>A hardware reset resets the 24MEN bit.</li> <li>A software reset <i>does not affect</i> the 24MEN bit.</li> </ul> <p>0 24M clock disabled 1 24M clock enabled</p>
9 FRR	<p>Free-Run or Restart mode.</p> <p>The FRR bit determines the behavior of the GPT when a compare event in channel 1 occurs.</p> <ul style="list-style-type: none"> <li>In Restart mode, after a compare event, the counter resets to 0x00000000 and resumes counting (after the occurrence of a compare event).</li> <li>In Free-Run mode, after a compare event, the counter continues counting until 0xFFFFFFFF and then rolls over to 0.</li> </ul> <p>0 Restart mode 1 Free-Run mode</p>
8–6 CLKSRC	<p>Clock Source select.</p> <p>The CLKSRC bits select which clock will go to the prescaler (and subsequently be used to run the GPT counter).</p> <ul style="list-style-type: none"> <li>The CLKSRC bit field value should only be changed after disabling the GPT by clearing the EN bit in this register (GPT_CR).</li> <li>A software reset does not affect the CLKSRC bit.</li> </ul> <p>000 No clock 001 Peripheral Clock 010 High Frequency Reference Clock 011 External Clock (CLKIN) 100 Low Frequency Reference Clock 101 Crystal oscillator as Reference Clock others Reserved</p>
5 STOPEN	<p>GPT Stop Mode enable.</p> <p>The STOPEN read/write control bit enables GPT operation <i>during Stop mode</i>.</p> <ul style="list-style-type: none"> <li>A hardware reset resets the STOPEN bit.</li> <li>A software reset <i>does not affect</i> the STOPEN bit.</li> </ul> <p>0 GPT is disabled in Stop mode. 1 GPT is enabled in Stop mode.</p>
4 DOZEEN	<p>GPT Doze Mode Enable.</p> <ul style="list-style-type: none"> <li>A hardware reset resets the DOZEEN bit.</li> <li>A software reset <i>does not affect</i> the DOZEEN bit.</li> </ul>

Table continues on the next page...

## GPT\_CR field descriptions (continued)

Field	Description
	0 GPT is disabled in doze mode. 1 GPT is enabled in doze mode.
3 WAITEN	GPT Wait Mode enable. The WAITEN read/write control bit enables GPT operation <i>during Wait mode</i> . <ul style="list-style-type: none"> <li>A hardware reset resets the WAITEN bit.</li> <li>A software reset <i>does not affect</i> the WAITEN bit.</li> </ul> 0 GPT is disabled in wait mode. 1 GPT is enabled in wait mode.
2 DBGEN	GPT debug mode enable. The DBGEN read/write control bit enables GPT operation <i>during Debug mode</i> . <ul style="list-style-type: none"> <li>A hardware reset resets the DBGEN bit.</li> <li>A software reset <i>does not affect</i> the DBGEN bit.</li> </ul> 0 GPT is disabled in debug mode. 1 GPT is enabled in debug mode.
1 ENMOD	GPT Enable mode. When the GPT is disabled (EN=0), then both the Main Counter and Prescaler Counter <i>freeze their current count values</i> . The ENMOD bit determines the value of the GPT counter when Counter is enabled again (if the EN bit is set). <ul style="list-style-type: none"> <li>If the ENMOD bit is 1, then the Main Counter and Prescaler Counter values are reset to 0 after GPT is enabled (EN=1).</li> <li>If the ENMOD bit is 0, then the Main Counter and Prescaler Counter restart counting <i>from their frozen values</i> after GPT is enabled (EN=1).</li> <li>If GPT is programmed to be disabled in a low power mode (STOP/WAIT), then the Main Counter and Prescaler Counter <i>freeze at their current count values</i> when the GPT enters low power mode.</li> <li>When GPT exits low power mode, the Main Counter and Prescaler Counter start counting from their frozen values, regardless of the ENMOD bit value.</li> <li>Setting the SWR bit will clear the Main Counter and Prescaler Counter values, regardless of the value of EN or ENMOD bits.</li> <li>A hardware reset resets the ENMOD bit.</li> <li>A software reset <i>does not affect</i> the ENMOD bit.</li> </ul> 0 GPT counter will retain its value when it is disabled. 1 GPT counter value is reset to 0 when it is disabled.
0 EN	GPT Enable. The EN bit is the GPT module enable bit. <b>Before setting the EN bit</b> , we recommend that <i>all registers be properly programmed</i> . <ul style="list-style-type: none"> <li>A hardware reset resets the EN bit.</li> <li>A software reset <i>does not affect</i> the EN bit.</li> </ul> 0 GPT is disabled. 1 GPT is enabled.

## 27.6.2 GPT Prescaler Register (GPT\_PR)

The GPT Prescaler Register (GPT\_PR) contains bits that determine the *divide value* of the clock that runs the counter.

Address: 209\_8000h base + 4h offset = 209\_8004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PRESCALER24M								PRESCALER							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### GPT\_PR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–12 PRESCALER24M	<p>Prescaler bits.</p> <p>24M crystal clock is divided by [PRESCALER24M + 1] before selected by the CLKSRC field. If 24M crystal clock is not selected, this feild takes no effect.</p> <p>0x0 Divide by 1            0x1 Divide by 2            ...            0xF Divide by 16</p>
11–0 PRESCALER	<p>Prescaler bits.</p> <p>The clock selected by the CLKSRC field is divided by [PRESCALER + 1], and then used to run the counter.</p> <ul style="list-style-type: none"> <li>A change in the value of the PRESCALER bits cause the Prescaler counter to reset and a new count period to start immediately.</li> <li>See <a href="#">Figure 27-3</a> for the timing diagram.</li> </ul> <p>0x000 Divide by 1            0x001 Divide by 2            ...            0xFFFF Divide by 4096</p>

## 27.6.3 GPT Status Register (GPT\_SR)

The GPT Status Register (GPT\_SR) contains bits that indicate that a counter has rolled over, and if any event has occurred on the Input Capture and Output Compare channels. The bits are cleared by writing a 1 to them.

Address: 209\_8000h base + 8h offset = 209\_8008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										ROV	IF2	IF1	OF3	OF2	OF1
W											w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**GPT\_SR field descriptions**

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value 0.
5 ROV	<p>Rollover Flag.</p> <p>The ROV bit indicates that the counter has reached its <i>maximum possible value</i> and <i>rolled over</i> to 0 (from which the counter continues counting). The ROV bit is only set if the counter has reached 0xFFFFFFFF in both Restart and Free-Run modes.</p> <p>0 Rollover has not occurred. 1 Rollover has occurred.</p>
4 IF2	<p>IF2 Input capture 2 Flag</p> <p>IF1 Input capture 1 Flag</p> <p>The IF<sub>n</sub> bit indicates that a capture event has occurred on Input Capture channel <i>n</i>.</p> <p>0 Capture event has not occurred. 1 Capture event has occurred.</p>
3 IF1	See IF2
2 OF3	<p>OF3 Output Compare 3 Flag</p> <p>OF2 Output Compare 2 Flag</p> <p>OF1 Output Compare 1 Flag</p> <p>The OF<sub>n</sub> bit indicates that a compare event has occurred on Output Compare channel <i>n</i>.</p>

*Table continues on the next page...*

**GPT\_SR field descriptions (continued)**

Field	Description
	0 Compare event has not occurred. 1 Compare event has occurred.
1 OF2	See OF3
0 OF1	See OF3

**27.6.4 GPT Interrupt Register (GPT\_IR)**

The GPT Interrupt Register (GPT\_IR) contains bits that control whether interrupts are generated after rollover, input capture and output compare events.

Address: 209\_8000h base + Ch offset = 209\_800Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**GPT\_IR field descriptions**

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value 0.
5 ROVIE	Rollover Interrupt Enable. The ROVIE bit controls the Rollover interrupt. 0 Rollover interrupt is disabled. 1 Rollover interrupt enabled.
4 IF2IE	IF2IE Input capture 2 Interrupt Enable IF1IE Input capture 1 Interrupt Enable The IFnIE bit controls the IFnIE Input Capture <i>n</i> Interrupt Enable.

*Table continues on the next page...*

**GPT\_IR field descriptions (continued)**

Field	Description
	0 IF2IE Input Capture <i>n</i> Interrupt Enable is disabled. 1 IF2IE Input Capture <i>n</i> Interrupt Enable is enabled.
3 IF1IE	See IF2IE
2 OF3IE	OF3IE Output Compare 3 Interrupt Enable OF2IE Output Compare 2 Interrupt Enable OF1IE Output Compare 1 Interrupt Enable The OF <i>n</i> IE bit controls the Output Compare Channel <i>n</i> interrupt.  0 Output Compare Channel <i>n</i> interrupt is disabled. 1 Output Compare Channel <i>n</i> interrupt is enabled.
1 OF2IE	See OF3IE
0 OF1IE	See OF3IE

**27.6.5 GPT Output Compare Register 1 (GPT\_OCR1)**

The GPT Compare Register 1 (GPT\_OCR1) holds the value that determines when a compare event will be generated on Output Compare Channel 1. Any write access to the Compare register of Channel 1 while in Restart mode (FRR=0) will reset the GPT counter.

An IP Bus Write access to the GPT Output Compare Register1 (GPT\_OCR1) occurs *after* one cycle of wait state; an IP Bus Read access occurs *immediately* (0 wait states).

Address: 209\_8000h base + 10h offset = 209\_8010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COMP																															
W	COMP																															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

**GPT\_OCR1 field descriptions**

Field	Description
31–0 COMP	Compare Value.  When the counter value equals the COMP bit field value, a compare event is generated on Output Compare Channel 1.

## 27.6.6 GPT Output Compare Register 2 (GPT\_OCR2)

The GPT Compare Register 2 (GPT\_OCR2) holds the value that determines when a compare event will be generated on Output Compare Channel 2.

Address: 209\_8000h base + 14h offset = 209\_8014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COMP																															
W																																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

**GPT\_OCR2 field descriptions**

Field	Description
31–0 COMP	Compare Value. When the counter value equals the COMP bit field value, a compare event is generated on Output Compare Channel 2.

## 27.6.7 GPT Output Compare Register 3 (GPT\_OCR3)

The GPT Compare Register 3 (GPT\_OCR3) holds the value that determines when a compare event will be generated on Output Compare Channel 3.

Address: 209\_8000h base + 18h offset = 209\_8018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COMP																															
W																																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

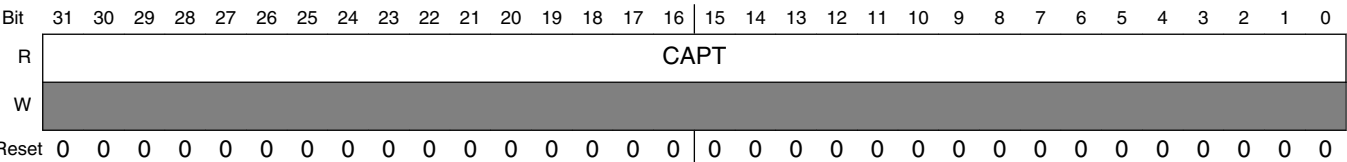
**GPT\_OCR3 field descriptions**

Field	Description
31–0 COMP	Compare Value. When the counter value equals the COMP bit field value, a compare event is generated on Output Compare Channel 3.

### 27.6.8 GPT Input Capture Register 1 (GPT\_ICR1)

The GPT Input Capture Register 1 (GPT\_ICR1) is a read-only register that holds the value *that was in the counter during the last capture event* on Input Capture Channel 1.

Address: 209\_8000h base + 1Ch offset = 209\_801Ch



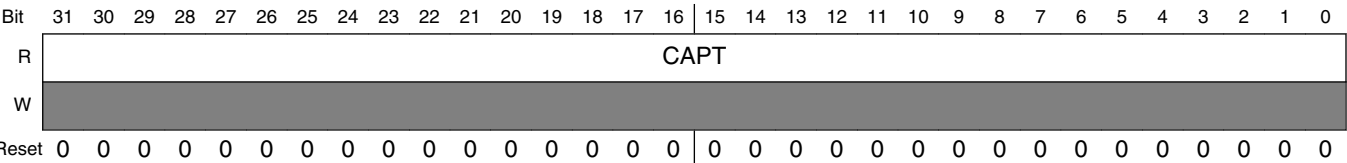
GPT\_ICR1 field descriptions

Field	Description
31–0 CAPT	Capture Value. After a capture event on Input Capture Channel 1 occurs, the current value of the counter is loaded into GPT Input Capture Register 1.

### 27.6.9 GPT Input Capture Register 2 (GPT\_ICR2)

The GPT Input capture Register 2 (GPT\_ICR2) is a read-only register which holds the value that was in the counter during the last capture event on input capture channel 2.

Address: 209\_8000h base + 20h offset = 209\_8020h



GPT\_ICR2 field descriptions

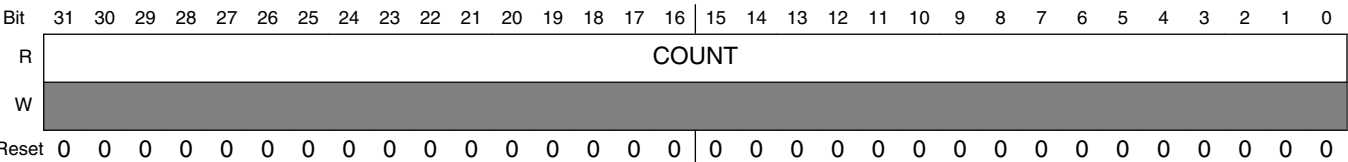
Field	Description
31–0 CAPT	Capture Value. After a capture event on Input Capture Channel 2 occurs, the current value of the counter is loaded into GPT Input Capture Register 2.



### 27.6.10 GPT Counter Register (GPT\_CNT)

The GPT Counter Register (GPT\_CNT) is the main counter's register. GPT\_CNT is a read-only register and can be read *without affecting the counting process* of the GPT.

Address: 209\_8000h base + 24h offset = 209\_8024h



GPT\_CNT field descriptions

Field	Description
31–0 COUNT	Counter Value. The COUNT bits show the current count value of the GPT counter.



# Chapter 28

## 2D Graphics Processing Unit (GPU2D)

### 28.1 Overview

The R2D GPU2D module is designed to display on a variety of consumer devices. Addressable screen sizes range from small displays featured on cell phones to large 1080p high definition displays.

The V2D GPU2D module is designed to display on a variety of consumer devices. Addressable screen sizes range from small displays featured on cell phones to large 1080p high definition displays.

The GPU2D cores provide powerful graphics at low power consumption, utilizing the smallest silicon footprints. Dynamic power consumption is minimized by extensive use of localized clock gating.

R2D GPU Hardware acceleration is brought to numerous 2D including graphical user interfaces (GUI), menu displays, flash animation, and gaming. The GPU R2D block diagrams is presented in the [Figure 28-1](#).

V2D GPU Hardware acceleration is brought to numerous VG applications including graphical user interfaces (GUI), menu displays, flash animation, and gaming. The GPU V2D block diagrams is presented in the [Figure 28-2](#).

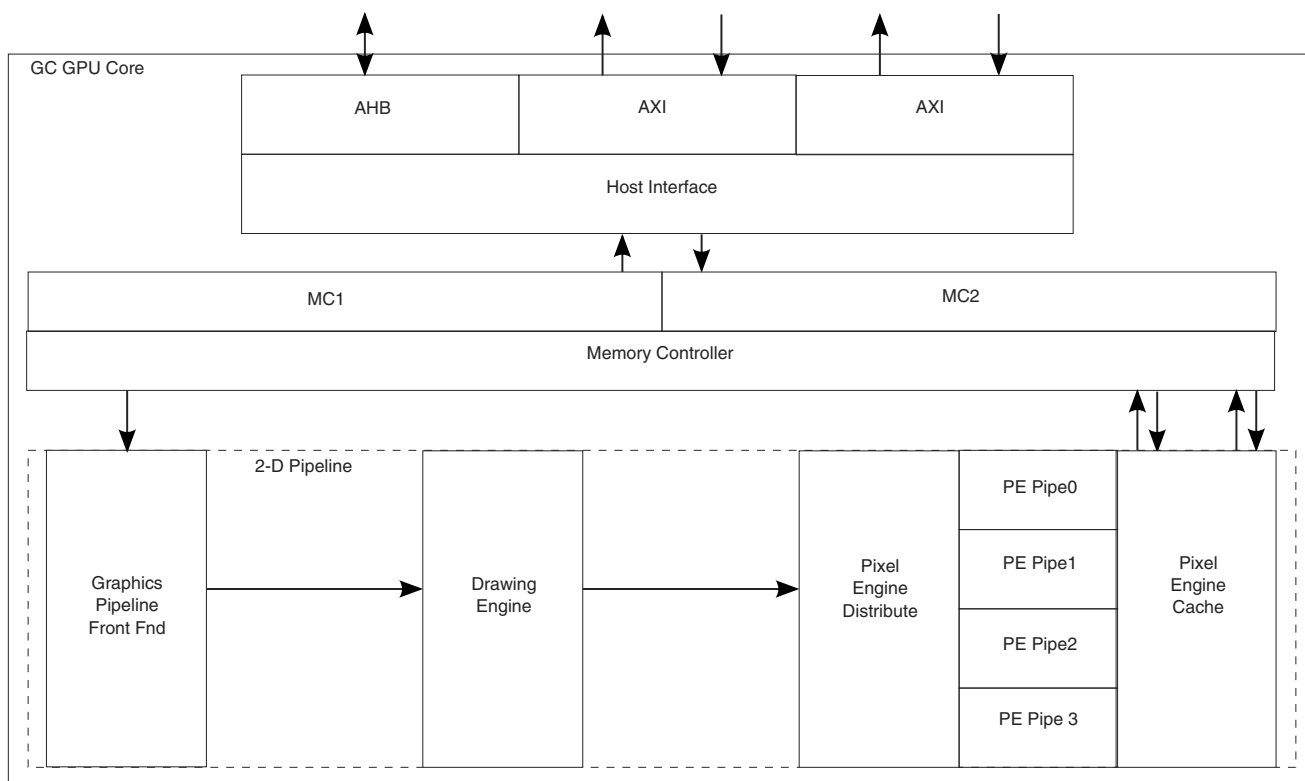
### 28.2 GPU2D Block Diagram

#### 28.2.1 R2D GPU

The R2D graphics processing unit (GPU) defines a high-performance 2D raster graphics core that accelerates the 2D graphics display.

R2D GPU supports acceleration of the following graphics APIs:

- DirectFB (on Linux)
- GDI/DirectDraw (on Windows CE)



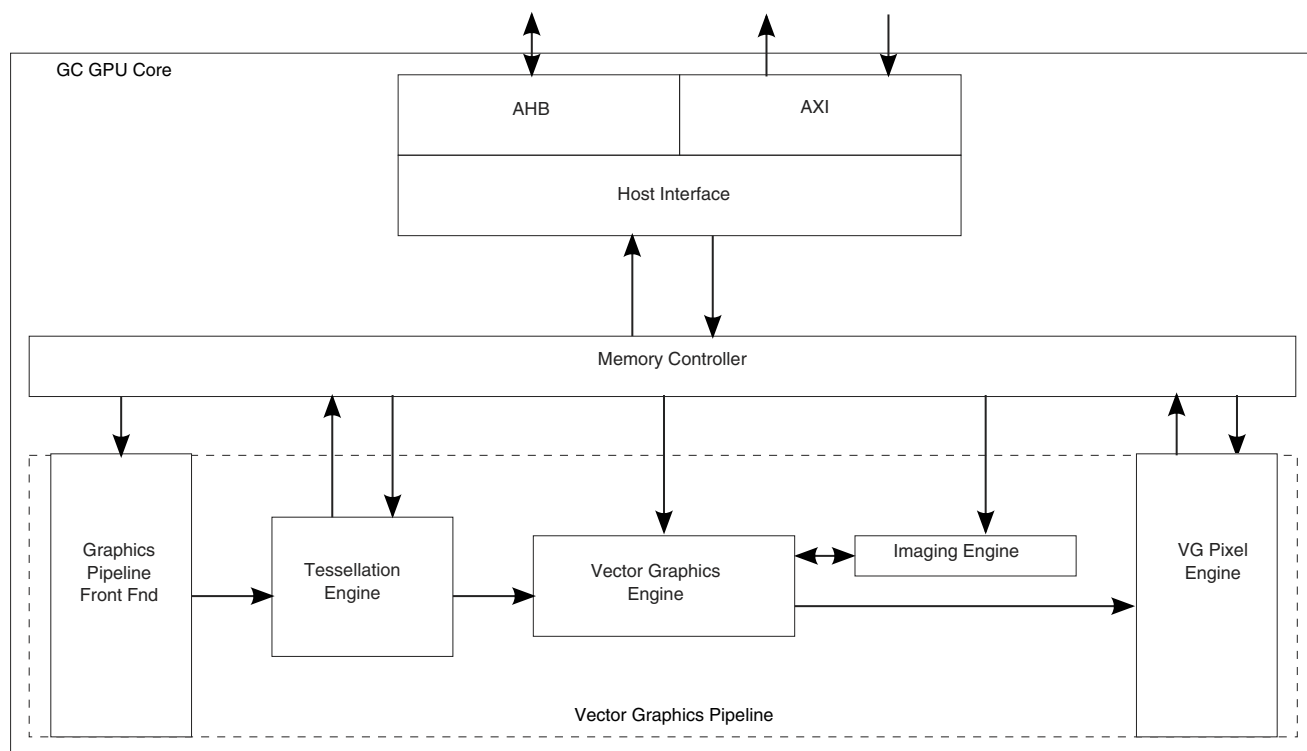
**Figure 28-1. R2D GPU Block Diagram**

### 28.2.2 V2D GPU

V2D GPU defines a high-performance graphics core designed for hardware acceleration of OpenVG vector graphics display. V2D GPU is designed for easy integration onto the SoC.

V2D GPU supports the following graphics APIs:

- OpenVG 1.1



**Figure 28-2. V2D GPU Block Diagram**

## 28.3 GPU2D Features

The following sections describe the functional features of the R2D and V2D GPU.

### 28.3.1 Full Featured R2D GPU Pipeline

- Bit BLT
- Stretch BLT
- Rectangle fill and clear
- Line drawing
- Filter BLT
- Mono expansion for text rendering
- ROP2, ROP3, and ROP4
- Alpha blending, including Java 2 Porter-Duff compositing blending rules
- 32K x 32K coordinate system
- 90 / 180 / 270 degree rotation
- Transparency by monochrome mask, chroma key, or pattern mask

### 28.3.2 Full Featured V2D GPU Pipeline

- Coordinate Systems and Transformations (Image drawing uses a 3x3 perspective transformation matrix)
- Viewport Clipping, Scissoring and Alpha Masking
- Paths
- Images
- Image Filters
- Paint(gradient and pattern)
- Blending
- Higher-level Geometric Primitives
- Image Warping

## 28.4 GPU2D OPERATIONS

### 28.4.1 R2D GPU Operations

Information detailing the R2D GPU operations can be found [here](#).

#### 28.4.1.1 Line

The LINE operation draws a line. Coordinates for two points are given: start point and end point. The end point is not drawn.

Lines are rendered using the Bresenham algorithm. The Bresenham algorithm has the advantage of using integer arithmetic and has no accumulation of rounding errors. In the case of line, only ROP2 and ROP4 are supported. It operates on pattern and destination. The pattern should have a transparency mask in order to use ROP4.

Clipping is supported for lines on a per pixel basis.

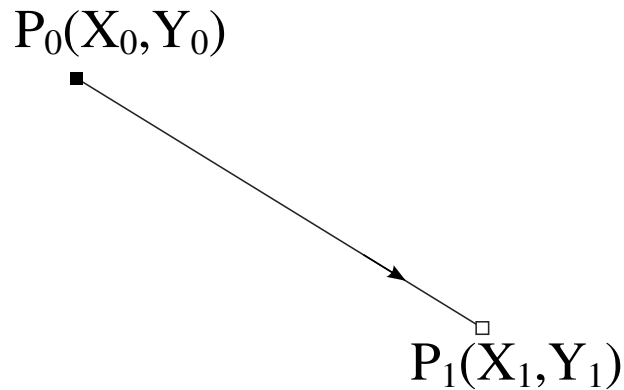


Figure 28-3. Line

### 28.4.1.2 Rectangle Fill and Clear

Rectangle fill creates a rectangle area with a given color. Essentially rectangle fill is a pattern fill, where an 8x8 pattern is initialized with the specified color. It supports ROP2 and ROP4 with the pattern and destination as its inputs. If ROP4 is used, the pattern should have a transparency mask.

Clear is similar to rectangle fill except that it does not use a pattern. A 32-bit clear value with 4-bit byte mask is used to fill the entire rectangle area.

Both rectangle fill and clear support clipping, which is performed on a per primitive basis.

### 28.4.1.3 BitBLT

Bit blit transfers data from one area of a memory (source) to another area of the memory (destination).

The source and destination can be from the same or different memory locations. Both source and destination must be described by a rectangular area. The source and destination rectangles can be the same size (most bit blits are of this nature) or they can be different sizes, in which case the operation becomes a stretch or shrink blit.

Bit blit supports ROP2, ROP3, and ROP4 which includes source, destination and pattern, and an optional transparency color.

Clipping can be performed on a primitive basis.

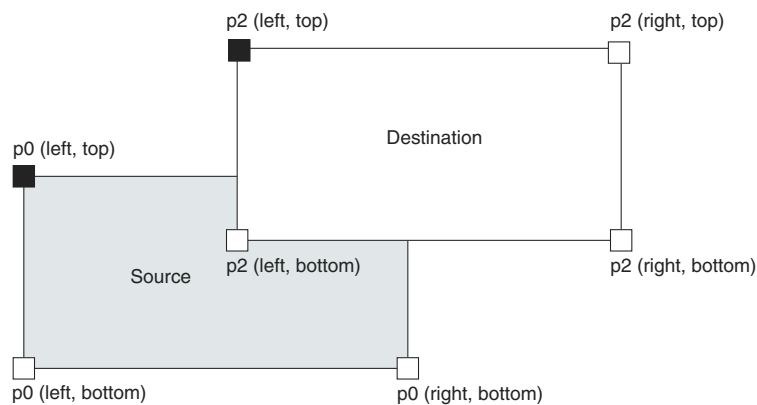


Figure 28-4. BitBLT

The BIT BLT primitive supports the following 10 source and 8 destination image formats:

Table 28-1. Bit BLT Formats

Formats	Source Image	Destination Image
A1R5G5B5	Yes	Yes
A4R4G4B4	Yes	Yes
X1R5G5B5	Yes	Yes
X4R4G4B4	Yes	Yes
R5G6B5	Yes	Yes
A8R8G8B8	Yes	Yes
X8R8G8B8	Yes	Yes
A8	Yes	Yes
1-bit monochrome	Yes	No
8-bit color index	Yes	No

28.4.1.4 Stretch BLT

The STRETCH BLT primitive performs a BitBlt operation with stretch or shrink. The modified Bresenham algorithm is used to generate corresponding coordinates for fast stretching.

The stretch factor is specified in a 15.0 fixed-point format. Stretch blit is not allowed to overlap therefore no part of source and destination can share any piece of memory. Non-stretch blits can overlap. For stretch blit, clipping is performed on a per pixel basis.



### 28.4.1.5 Monochrome Expansion and Mask BLT

Monochrome expansion and mask blit are different operations, although both use the bit stream from command buffer and both can be the source for ROP4 source selection. This means that each output pixel can be a combination of source, pattern, monochrome mask (for masked blits) and destination.

#### 28.4.1.5.1 Monochrome expansion

For monochrome expansion, the bit from the stream is used to switch on/off a solid color that is defined in a register. This mechanism enables the use of just one bit per pixel to represent colors. In effect, the MONO EXPANSION primitive increases color representation from one bit per pixel to multiple bits per pixel. A typical application for mono color is font drawing.

Monochrome expansion does not support overlapping of the source and destination. It is the responsibility of the driver to make sure that the command will never be executed with overlapping source and destination.

#### 28.4.1.5.2 Mask BLT

For Mask BLT, the bit from the stream is used to toggle on/off a color in the source frame buffer. Mask BLT takes its color source from memory and its monochrome mask from the command stream. Clipping is supported and is performed on a per pixel basis.

#### 28.4.1.6 Filter BLT

Filter blit performs high quality scaling, up or down, using an FIR re-sampling filter with up to 9 taps. Sub-pixel coordinates (locations between the pixel grids) are generated by the drawing engine. The filter block in the drawing engine uses the sub-pixel information to select the appropriate filter kernel. R2D GPU processes 1 pixel every cycle when performing filter blit.

A stretch- or shrink-factor of 15.16 fixed-point format is supported. To generate a single destination pixel requires 9 source pixels. An image is scaled in two passes, one for X-dimension (HOR\_FILTER\_BLT) and the other for Y-dimension (VER\_FILTER\_BLT). Software sets up the filter kernel/coefficient table and the kernel size, as well as a temporary buffer for storing intermediate results. After the first pass is completed, intermediate results are sent back to memory, and then the second pass starts to scale the first-pass image. Because of this two-step procedure, the throughput of FILTER BLT is lower than that of STRETCH BLT. Also the Filter Kernel Table may need to be reloaded, and some cycles are consumed in calculating the stepping parameters.

When the stretch or shrink factor is 1, the filterBlit works as a bitBlit copy. It can be used as format converter in that case, for instance, YUV to RGB converter. To use as a format converter, only one pass (HOR\_FILTER\_BLT or VER\_FILTER\_BLT) is needed. To optimize the memory bandwidth, when using filterBlit to do YUV to RGB filtering, the temporary target buffer format can be specified as YUY2 to process Y-dimension filtering (VER\_FILTER\_BLT). This is to avoid converting YUV to A8R8G8B8 in the 1st vertical pass to reduce the memory bandwidth and increase the pixel processing rate. This is the only special case that GPU may use YUY2 as target format.

The FILTER BLT primitive supports the following 13 source and 7 destination image formats:

Filter BLT Formats

Formats		Source Image	Destination Image
A1R5G5B5		Yes	Yes
A4R4G4B4		Yes	Yes
A8R8G8B8		Yes	Yes
R5G6B5		Yes	Yes
X1R5G5B5		Yes	Yes
X4R4G4B4		Yes	Yes
X8R8G8B8		Yes	Yes
YUV	NV12 (4:2:0, 2 planes)	Yes	No
	NV16 (4:2:2, 2 planes)	Yes	No
	UYVY (4:2:2, interleave)	Yes	No
	YUY2 (4:2:2, interleave)	Yes	No
	YV12 (4:2:0, 3 planes)	Yes	No
	8-bit color index	Yes	No

### 28.4.1.7 R2D Performance of different operations

Table 28-2. Performance of different operations without rotation

Primitive	Peak Performance	Source/destination overlap	Clipping
Line	1 pixel / cycle	N/A	Done on pixel basis
Rectangle	2.4 pixel / cycle	N/A	Done on primitive basis
Clear	2.4 pixel / cycle	N/A	Done on primitive basis
Blit	2.4 pixel / cycle	Overlap is allowed	Done on primitive basis
Stretch blit	1 pixel / cycle	No overlap allowed	Done on pixel basis
Monochrome expansion	1.6 pixel / cycle	No overlap allowed	Done on pixel basis
Filter blit	1 pixel / cycle	N/A	N/A

**Table 28-3. Performance of different operations with rotation**

Primitive	Performance	Source/destination overlap	Clipping
Line	1 pixel / cycle	N/A	Done on pixel basis
Rectangle	2 pixel / cycle	N/A	Done on primitive basis
Clear	2 pixel / cycle	N/A	Done on primitive basis
Blit	2 pixel / cycle	Overlap is allowed	Done on primitive basis
Stretch blit	1 pixel / cycle	No overlap allowed	Done on pixel basis
Monochrome expansion	1 pixel / cycle	No overlap allowed	Done on pixel basis
Filter blit	1 pixel / cycle	N/A	N/A

### 28.4.1.8 Rotation

90° / 180° / 270° / X-Flip / Y-Flip / Mirror rotation is supported for all primitives.

### 28.4.1.9 Transparency Mode

For monochrome expansion:

- Opaque
- Conditional transparency. Transparent if the current pixel matches the specified value.

For blits:

- Opaque
- Masked transparency. Transparent if the mask for the current pixel or pattern is zero.
- Source Conditional transparency. Transparent if the source pixel is within the specified value range.
- Destination Conditional transparency. Transparent if the destination pixel is not within the specified value range.

### 28.4.1.10 Clipping

One clipping rectangle is supported for all bitBlit primitives.

### 28.4.1.11 R2D GPU Data Formats

The graphics engine supports 14 source data formats. In addition to these 14 source formats, for RGB source formats, GPU also supports their swizzle formats (ARGB, RGBA, ABGR, BGRA) for RGB formats. For YUV formats, GPU supports their U/V swap formats.

- A1R5G5B5
- A4R4G4B4
- A8R8G8B8
- R5G6B5
- X1R5G5B5
- X4R4G4B4
- X8R8G8B8
- A8
- NV12
- NV16
- UYVY (4:2:2)
- YUY2 (4:2:2)
- YV12 (4:2:0)
- 8-bit color index

There are 8 destination data formats supported by the graphics engine. In addition to these destination RGB formats, their swizzle formats (ARGB, RGBA, ABGR, BGRA) are also supported.

- A1R5G5B5
- A4R4G4B4
- A8R8G8B8
- R5G6B5
- X1R5G5B5
- X4R4G4B4
- X8R8G8B8
- A8

### 28.4.1.12 ARGB Data Conversion of R2D GPU

The pixels read from source or destination will be expanded into A8R8G8B8 format internally to maintain lossless pixel operations. The resulting pixels will be converted into the destination format.

### 28.4.1.13 YUV to RGB Conversion of R2D GPU

YUV data can be converted into 8-bit per component RGB format at the output of the cache only. Once converted, there is no way back to YUV format. GPU supports BT.601 and BT.709 YUV to RGB color conversion standards.

In BT.601, the YUV to RGB conversion is done using the following approximation:

$$16 \leq Y \leq 235$$

$$16 \leq U \leq 240$$

$$16 \leq V \leq 240$$

$$A = Y - 16$$

$$B = U - 128$$

$$C = V - 128$$

$$R = \text{clip}((298 * A + 410 * C + 128) \gg 8)$$

$$G = \text{clip}((298 * A - 101 * B - 209 * C + 128) \gg 8)$$

$$B = \text{clip}((298 * A + 519 * B + 128) \gg 8)$$

The Y, U and V components are clamped prior to the conversion.

Y is clamped between 16 and 235, inclusively.

U and V are clamped between 16 and 240, inclusively.

In BT.709, the R, G, B equations are slightly changed to

$$R = \text{clip}((298 * A + 461 * C + 128) \gg 8)$$

$$G = \text{clip}((298 * A - 55 * B - 137 * C + 128) \gg 8)$$

$$B = \text{clip}((298 * A + 543 * B + 128) \gg 8)$$

### 28.4.1.14 Color Index Input Conversion Support of R2D GPU

Color index is supported for source data only. A look-up table with 256 entries is provided for indexing the data. The table is fully programmable. The conversion is done when pixels are read out of the cache.

### 28.4.1.15 Source/Destination Pre-multiply and De-Multiply Support

GPU supports source pre-multiply source alpha or global alpha, or source global color for global colorizing. On destination, the GPU supports destination pre-multiply destination alpha, destination de-multiply alpha.

### 28.4.1.16 Alpha Blending

The GPU supports alpha blending together with ROP. The alpha blending function is performed on ROP function result source.

The general alpha blending equations are:

$$Cd = Fs * Cs' + Fd * Cd'$$

$$Ad = Fs * As'' + Fd * Ad''$$

Where

- Cs' is the source color component (adjusted for NPM if necessary)
- Cd' is the destination color component (adjusted for NPM if necessary)
- As'' is the modified source alpha component
- Ad'' is the modified destination alpha component
- Fs is fraction of the source that contributes to the final value
- Fd is fraction of the destination that contributes to the final value

The blending is done in 5 logical stages (not real implementation stages):

1. Transparent/opaque conversion
  - In this stage, the incoming alpha (source or destination independently) can be inverted if needed to match the internal alpha rule. Internally, an alpha of 0 means transparent, while an alpha of "0xFF" means opaque. External content might follow the opposite rule. The output of the block is either As (Ad for destination) or 1-As (1-Ad for destination).
2. Global value substitution
  - A global alpha value from a register can be used to substitute or scale the incoming alpha. An incoming alpha As can pass-through, be directly substituted by Ags (global alpha) or scaled by the global alpha value (As \* Ags). The source and destination have distinct global alpha values.
3. Blending factor generation
  - At this stages, the blending factors are generated (refer to table below). Each alpha can take the values 0, 1, A or 1-A depending on the blending mode.
4. Final blending
  - This is the final stage which implements blending equations.

The fractions take the values described in the following table, depending on the blending mode.

**Table 28-4. Blending Modes Fractions Description**

Blending Mode	Fs	Fd
Clear	0	0
SRC	1	0
DST	0	1
SRC_OVER	1	$1 - A_s''$
DST_OVER	$1 - A_d''$	1
SRC_IN	$A_d''$	0
DST_IN	0	$A_s''$
SRC_OUT	$1 - A_d''$	0
DST_OUT	0	$1 - A_s''$
SRC_ATOP	$A_d''$	$1 - A_s''$
DST_ATOP	$1 - A_d''$	$A_s''$
XOR	$1 - A_d''$	$1 - A_s''$

To control the blending modes, the following register fields are used:

- 1 bit for transparent/opaque conversion for source alpha
- 1 bit for transparent/opaque conversion for destination alpha
- 2 bits for source alpha modifications, to specify the 3 cases ( $A_s$ ,  $A_g$ ,  $A_s * A_g$ )
- 2 bits for destination alpha modifications, to specify the 3 cases ( $A_d$ ,  $A_g$ ,  $A_d * A_g$ )
- 4 bits to select between the 12 blending modes
- 8-bits for global source alpha
- 8-bits for global destination alpha

Alpha blending is supported on bit blit and filter blit primitives.

### **28.4.1.17 GPU Cache Management**

SW cache flush is supported to flush the GPU cache to memory. Auto-flush of GPU cache is also supported. SW sets up the auto-flush interval, and HW will do the cache flush automatically at programmable intervals.

## **28.4.2 V2D GPU Operations**

Information detailing the V2D GPU operations can be found [here](#).

### **28.4.2.1 OPENVG 1.1-API STANDARD for VECTOR GRAPHICS ACCELERATION**

OpenVG is a royalty-free, cross-platform API managed by the member-funded consortium known as Khronos Group. It provides a low-level hardware acceleration interface for vector graphics libraries such as Flash and SVG. OpenVG is used for acceleration of high-quality vector graphics for user interfaces and text on small screen devices.

#### **28.4.2.2 Advantages of Using OpenVG**

- Hardware accelerators can reduce power consumption by up to 90% compared to a software engine.
- Scalability with high-quality rendering, including anti-aliasing, to different screen sizes without multiple bitmaps.

#### **28.4.2.3 OpenVG Target Applications**

- SVG Viewers
- Portable Mapping Applications
- E-book Readers
- Games
- Scalable User Interface

#### **28.4.2.4 OpenVG Features**



#### **28.4.2.4.1 Core API**

- Coordinate Systems and Transformations (Image drawing uses a 3x3 perspective transformation matrix)
- Viewport Clipping, Scissoring and Alpha Masking
- Paths
- Images
- Image Filters
- Paint (gradient and pattern)
- Blending
- Dithering

#### **28.4.2.4.2 The VGU Utility Library**

- Higher-level Geometric Primitives
- Image Warping

#### **28.4.2.4.3 OpenVG Rendering Pipeline**

- Stage 1: Path, Transformation, Stroke, and Paint
- Stage 2: Stroked Path Generation
- Stage 3: Transformation
- Stage 4: Rasterization
- Stage 5: Clipping and Masking
- Stage 6: Paint Generation
- Stage 7: Image Interpolation
- Stage 8: Blending and Anti-aliasing



# Chapter 29

## I2C Controller (I2C)

### 29.1 Overview

This chapter describes block-level operation and programming of I2C. The chapter is intended for a block-driver software developer. To understand how the block is integrated at the SoC level, a system software developer should see discussions of the block in the appropriate SoC-level chapter(s).

**References:** This document assumes an understanding of the following document:

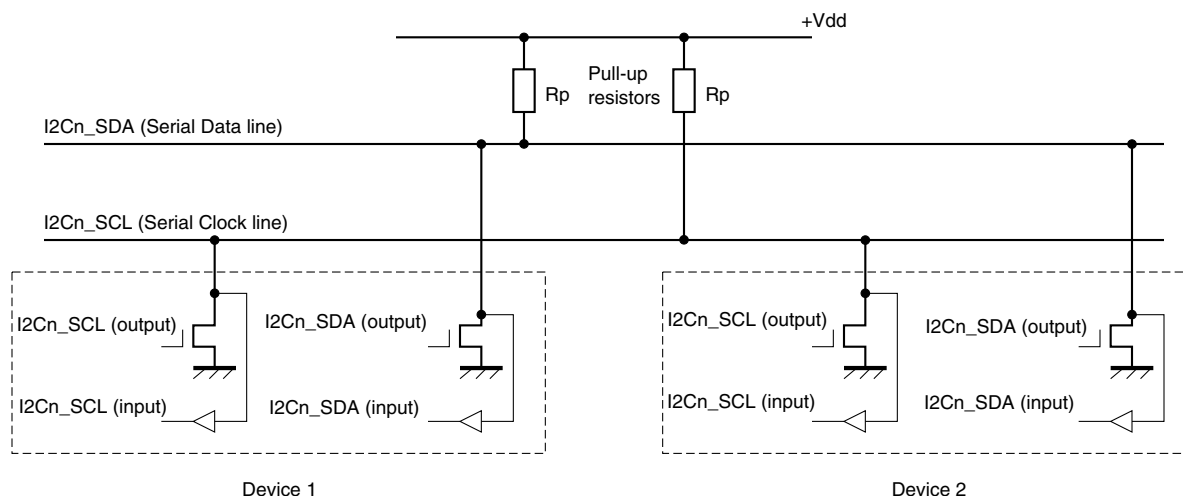
- *The I2C Bus Specification*, Version 2.1, by Philips Semiconductor

The Inter IC (I2C) provides functionality of a standard I2C slave and master. The I2C is designed to be compatible with the standard NXP I2C bus protocol.

#### NOTE

Three independent I2C channels are available.

I2C is a two-wire, bidirectional serial bus that provides a simple, efficient method of data exchange, minimizing the interconnection between devices. This bus is suitable for applications requiring occasional communications over a short distance between many devices. The flexible I2C standard allows additional devices to be connected to the bus for expansion and system development. See the connection diagram in the figure below.



**Figure 29-1. Connection of devices to I2C bus**

The I2C interface operates at up to 400 kbps, but the speed is dependent on the I2C bus loading and timing characteristics. For pin requirement details, see *The I2C Bus Specification*. The I2C system is a true multimaster bus including arbitration and collision detection that prevents data corruption if multiple devices attempt to control the bus simultaneously. This feature supports complex applications with multiprocessor control and can be used for rapid testing and alignment of end products through external connections to an assembly-line computer. The figure below shows the block diagram of I2C.

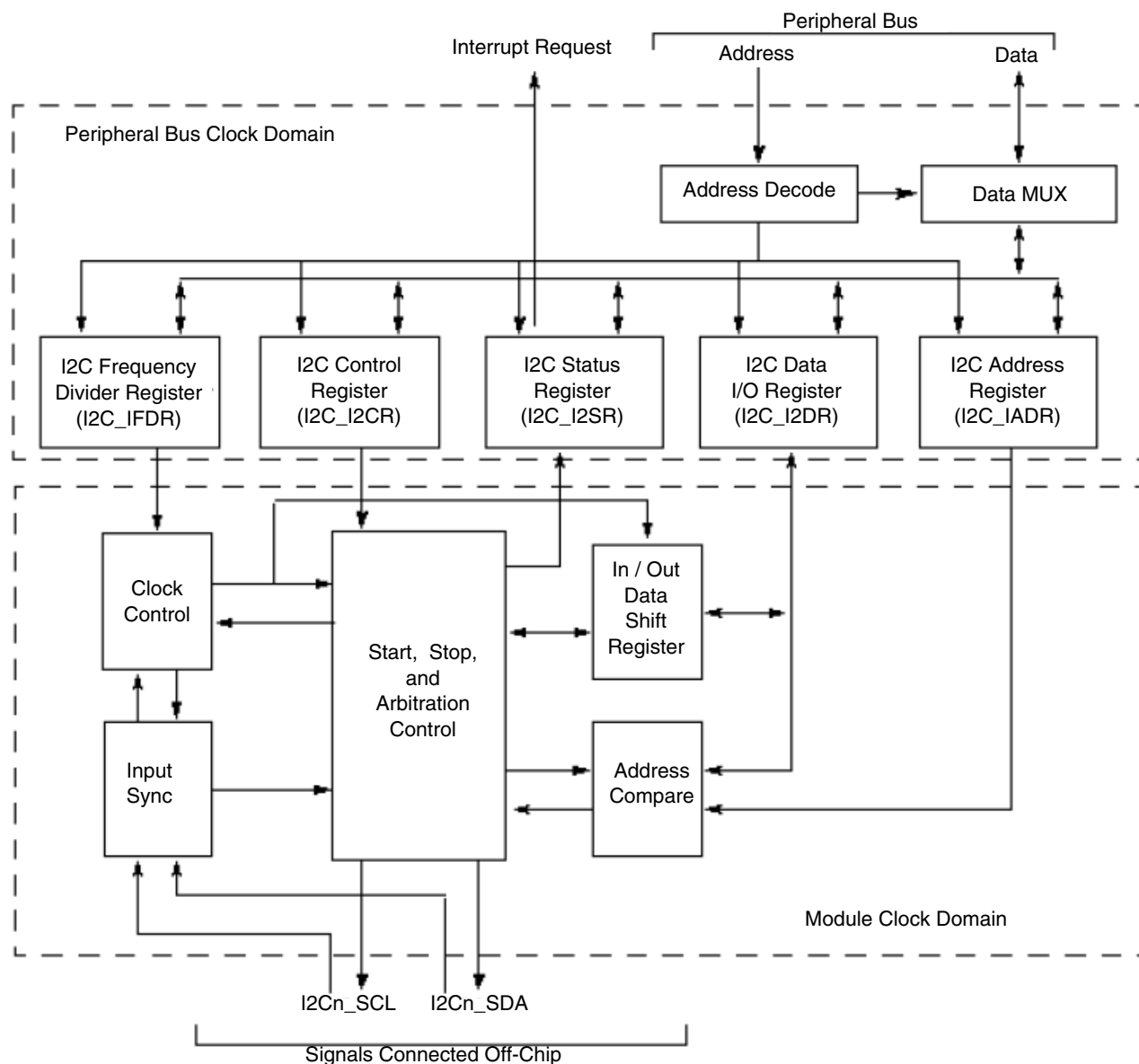


Figure 29-2. I2C block diagram

### 29.1.1 Features

The I2C has the following key features:

- Compatibility with I2C bus standard
- Multimaster operation
- Software programmability for one of 64 different serial clock frequencies
- Software-selectable acknowledge bit
- Interrupt-driven, byte-by-byte data transfer
- Arbitration-lost interrupt with automatic mode switching from master to slave

- Calling address identification interrupt
- Start and stop signal generation/detection
- Repeated Start signal generation
- Acknowledge bit generation/detection
- Bus-busy detection

### 29.1.2 Modes and operations

The I2C operates primarily in two functional modes: Standard mode and Fast mode.

- In Standard mode, I2C supports the data transfer rates up to 100 kbits/s.
- In Fast mode, data transfer rates up to 400 kbits/s can be achieved. Per block operation, there is no special configuration required for Fast or Standard mode. It is the data transfer rate that distinguishes Standard and Fast mode.

## 29.2 External Signals

This section discusses I2C signals that connect off-chip.

For I2C compliance, all devices connected to the I2Cn\_SCL and I2Cn\_SDA signals must have open-drain or open-collector outputs. The logic AND function is implemented on both lines with external pull-up resistors.

Inputs of I2Cn\_SCL and I2Cn\_SDA also need to be manually enabled by setting the SION bit in the IOMUX after the corresponding PADs are selected as I2C function.

The table below describes all I2C signals that connect off-chip.

**Table 29-1. I2C External Signals**

I2C1_SCL (SCL)	Serial Clock	AUD_RXFS	ALT1	IO
		I2C1_SCL	ALT0	
		HSIC_DAT	ALT1	
I2C1_SDA (SDA)	Serial Data	AUD_RXC	ALT1	IO
		I2C1_SDA	ALT0	
		HSIC_STROBE	ALT1	
I2C2_SCL (SCL)	Serial Clock	EPDC_SDCLK	ALT2	IO
		I2C2_SCL	ALT0	
		KEY_COL0	ALT1	
		LCD_DAT16	ALT4	

*Table continues on the next page...*

**Table 29-1. I2C External Signals (continued)**

I2C2_SDA (SDA)	Serial Data	EPDC_SDLE	ALT2	IO
		I2C2_SDA (SDA)	ALT0	
		KEY_ROW0	ALT1	
		LCD_DAT17	ALT4	
I2C3_SCL (SCL)	Serial Clock	AUD_RXFS	ALT4	IO
		EPDC_SDCE2	ALT1	
		REF_CLK_24M	ALT1	
I2C3_SDA (SDA)	Serial Data	AUD_RXC	ALT4	IO
		EPDC_SDCE3	ALT1	
		REF_CLK_32K	ALT1	

## 29.3 Clocks

There are two input clocks for I2C.

The following table describes the clock sources for I2C. Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 29-2. I2C Clocks**

Clock name	Clock Root	Description
ipg_clk_patref	perclk_clk_root	Module clock
ipg_clk_s	perclk_clk_root	Peripheral access clock

- Peripheral clock (ipg\_clk\_s): This clock is used for peripheral bus register read/writes.
- Module clock (ipg\_clk\_patref): This is the functional clock of the I2C. The serial bit clock frequency is derived from the module clock. The module clock and peripheral clocks are synchronous with each other. The minimum frequency of the module clock should be 12.8 MHz for Fast mode to achieve 400-kbps operation.

## 29.4 Functional description

This section provides a complete functional description of the block.

## 29.4.1 I2C system configuration

After a reset, the I2C defaults to Slave Receive operations. Thus, when not operating as a master or responding to a slave transmit address, the I2C defaults to the Slave Receive state.

For exceptions, see [Initialization sequence](#).

### NOTE

The I2C is designed to be compatible with the Philips™ I2C bus protocol. For information on system configuration, protocol, and restrictions, see the *I2C Bus Specification*, version 2.1, by Philips Semiconductors. The I2C supports Standard and Fast modes only.

## 29.4.2 Arbitration procedure

If multiple devices simultaneously request the bus, the bus clock is determined by a synchronization procedure in which the low period equals the longest clock-low period among the devices, and the high period equals the shortest. A data arbitration procedure determines the relative priority of competing devices.

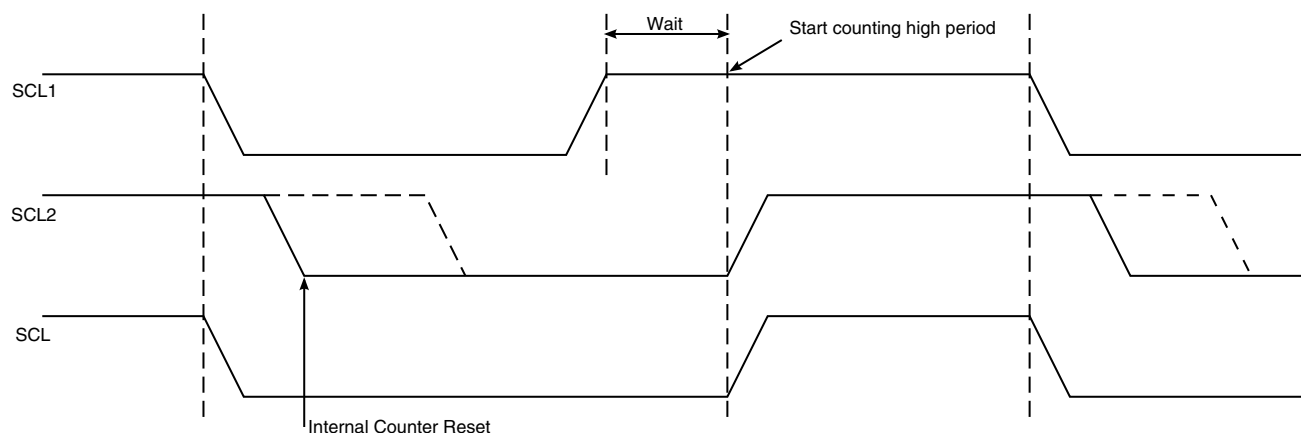
A device loses arbitration if it sends logic high while another sends logic low; it immediately switches to Slave Receive mode and stops driving I2Cn\_SDA. In this case, the transition from master to Slave mode does not generate a Stop condition. Meanwhile, hardware sets the arbitration lost bit in the I2C Status register (I2C\_I2SR[IAL] to indicate loss of arbitration).



### 29.4.3 Clock synchronization

Because wire-AND logic is used, a high-to-low transition on SCL affects devices connected to the bus. Devices start counting their low period when the master drives SCL low. When a device clock goes low, it holds SCL low until the Clock High state is reached. However, the low-to-high change in this device clock may not change the state of SCL if another device clock is still in its low period. Therefore, the device with the longest low period holds the synchronized clock SCL low.

Devices with shorter low periods enter a High Wait state during this time (see [Figure 29-3](#)). When all devices involved have counted off their low periods, the synchronized clock SCL is released and pulled high. There is then no difference between device clocks and the state of SCL, so all of the devices start counting their high periods. The first device to complete its high period pulls SCL low again.



**Figure 29-3. Synchronized clock SCL**

### 29.4.4 Handshaking

The clock synchronization mechanism can be used as a handshake in data transfers. Slave devices can hold SCL low after completing one byte transfer (9 bits). In such a case, the clock mechanism halts the bus clock and forces the master clock into a Wait state until the slave releases SCL.

## 29.4.5 Clock stretching

Slaves can use the clock synchronization mechanism to slow down the transfer bit rate. After the master has driven SCL low, the slave can drive SCL low for the required period and then release it. If the slave SCL low period is longer than the master SCL low period, the resulting SCL bus signal low period is stretched.

## 29.4.6 Peripheral bus accesses

I2C is a 16-bit block. Only half-word accesses should be performed to the block.

## 29.4.7 Generation of transfer error on IP bus

If an address is received on the peripheral slave bus interface but it is not implemented, an access error is generated.

## 29.4.8 Reset

The I2C can be reset in the following ways:

- Global reset: A hard asynchronous reset of the whole I2C
- Software reset: An internal reset for the whole I2C (except for I2C\_IADR and I2C\_IFDR registers) initiated by deasserting the I2C\_I2CR[IEN] bit

## 29.4.9 Interrupts

There is only one interrupt from the block, which is enabled by setting the I2C\_I2CR[IIEN] bit.

The interrupt is generated in any one of the following conditions:

- One byte transfer is completed (the interrupt is set at the falling edge of the ninth clock).
- An address is received that matches its own specific address in Slave Receive mode.
- Arbitration is lost.

## 29.4.10 Byte order

The block only supports the Little-Endian mode.

## 29.5 Initialization

### NOTE

Ensure the input select pins for IOMUXC are configured correctly for I2C.

### 29.5.1 Initialization sequence

Before the interface can transfer serial data, registers must be initialized, as listed here.

1. Set the data sampling rate (I2C\_IFDR[IC] to obtain SCL frequency from the system bus clock.
2. Update the address in the (I2C\_IADR) to define its slave address (address can range from 0 to 0x7f).
3. Set the I2C enable bit (I2C\_I2CR[IEN]) to enable the I2C bus interface system.
4. Modify the bits in the I2C\_I2CR to select Master/Slave mode, Transmit/Receive mode, and Interrupt-Enable or not.

### 29.5.2 Generation of Start

After completion of the initialization procedure, serial data can be transmitted by selecting the Master Transmit mode. On a multimaster bus system, the busy bus (I2C\_I2SR[IBB]) must be tested to determine whether the serial bus is free. If the bus is free (IBB = 0), the Start signal and the first byte (the slave address) can be sent. The data written to the data register comprises the address of the desired slave and the LSB indicates the transfer direction.

The free time between a Stop and the next Start condition is built into the hardware that generates the Start cycle. Depending on the relative frequencies of the system clock and the SCL period, it may be necessary to wait until the I2C is busy after writing the calling address to the data register (I2C\_I2DR), before proceeding to load data into the data register (I2C\_I2DR).

### 29.5.3 Post-transfer software response

Sending or receiving a byte sets the data transferring bit (I2C\_I2SR[ICF]), which indicates one byte of communication is finished. Upon completion, the interrupt status (I2C\_I2SR[IIF]) is also set. An external interrupt is generated if the interrupt enable (I2C\_I2CR[IIEN]) is set. The software must first clear the interrupt status (I2C\_I2SR[IIF]) in the interrupt routine.

See the flow chart in [Figure 29-5](#).

The data transferring bit (I2C\_I2SR[ICF]) is cleared either by reading from I2C\_I2DR in Receive mode or by writing to this register in Transmit mode.

The software can service the I2C I/O in the main program by monitoring the interrupt status (I2C\_I2SR[IIF]) if the interrupt enable is deasserted. In this case, the interrupt status should be polled in the data transferring bit (I2C\_I2SR[ICF]) because the operation is different when arbitration is lost.

When an interrupt occurs at the end of the address cycle, the master is always in Transmit mode; that is, the address is sent. If Master Receive mode is required, then I2C\_I2CR[MTX] should be toggled and a dummy read of the I2C\_I2DR register must be executed to trigger receive data.

During Slave-mode address cycles (I2C\_I2SR[IAAS] = 1), the slave read/write bit I2C\_I2SR[SRW] is read to determine the direction of the next transfer. The transmit/receive bit (I2C\_I2CR[MTX]) should also be programmed accordingly. For Slave-mode data cycles (IAAS = 0), SRW is invalid. MTX should be read to determine the current transfer direction.

### 29.5.4 Generation of Stop

A data transfer ends when the master signals a Stop, which can occur after all data is sent.

For a master receiver to terminate a data transfer, it must inform the slave transmitter by not acknowledging the last data byte. This is done by setting the transmit acknowledge bit (I2C\_I2CR[TXAK]) before reading the next-to-last byte. Before the last byte is read, a Stop signal must be generated.

### 29.5.5 Generation of Repeated Start

After the data transfer, if the master still requires the bus, it can signal another Start followed by another slave address without signaling a Stop.

### 29.5.6 Slave mode

In the slave interrupt service routine (see [Figure 29-5](#)), the block addressed as slave bit (IAAS) should be tested to check if a calling of its own address has just been received. If IAAS is set, software should set the Transmit/Receive mode select bit (I2C\_I2CR[MTX]) according to the I2C\_I2SR[SRW]. Writing to the I2C\_I2CR clears the IAAS automatically. The only time IAAS is read as set is from the interrupt at the end of the address cycle where an address match occurred; interrupts resulting from subsequent data transfers will have IAAS cleared. A data transfer can now be initiated by writing information to I2C\_I2DR for slave transmits, or read from I2C\_I2DR in Slave Receive mode. A dummy read of I2C\_I2DR in Slave Receive mode releases SCL, allowing the master to send data.

In the slave transmitter routine, the receive acknowledge bit (I2C\_I2SR[RXAK]) must be tested before sending the next byte of data. Setting RXAK means an end-of-data signal from the master receiver, after which the software must switch it from Transmit to Receiver mode. Reading the data register (I2C\_I2DR) then releases SCL so the master can generate a Stop signal.

### 29.5.7 Arbitration lost

If several devices try to engage the bus at the same time, one becomes master. Hardware immediately switches devices that lose arbitration to Slave Receive mode. Data output to 12Cn\_SDA stops, but 12Cn\_SCL is still generated until the end of the byte during which arbitration is lost. An interrupt occurs at the falling edge of the ninth clock of this transfer if the arbitration is lost (I2C\_I2SR[IAL] = 1), and the Slave mode is selected (I2C\_I2CR[MSTA] = 0).

See the flow chart in [Figure 29-5](#).

If a device that is not a master tries to transmit or do a Start, hardware inhibits the transmission, clears MSTA without signaling a Stop, generates an interrupt to the ARM platform, and sets I2C\_I2SR[IAL] to indicate a failed attempt to engage the bus. When considering these cases, the slave service routine should first test I2C\_I2SR[IAL], and the software should clear it if it is set.

For Multimaster mode, when an I2C is enabled when the bus is busy and asserts Start, the I2C\_I2SR[IAL] bit gets set only for 12Cn\_SDA=0, 12Cn\_SCL=0/1, 12Cn\_SDA=1, and 12Cn\_SCL=0; but not for 12Cn\_SDA=1 and I2Cn\_SCL=1, which is the equivalent of Bus Idle state.

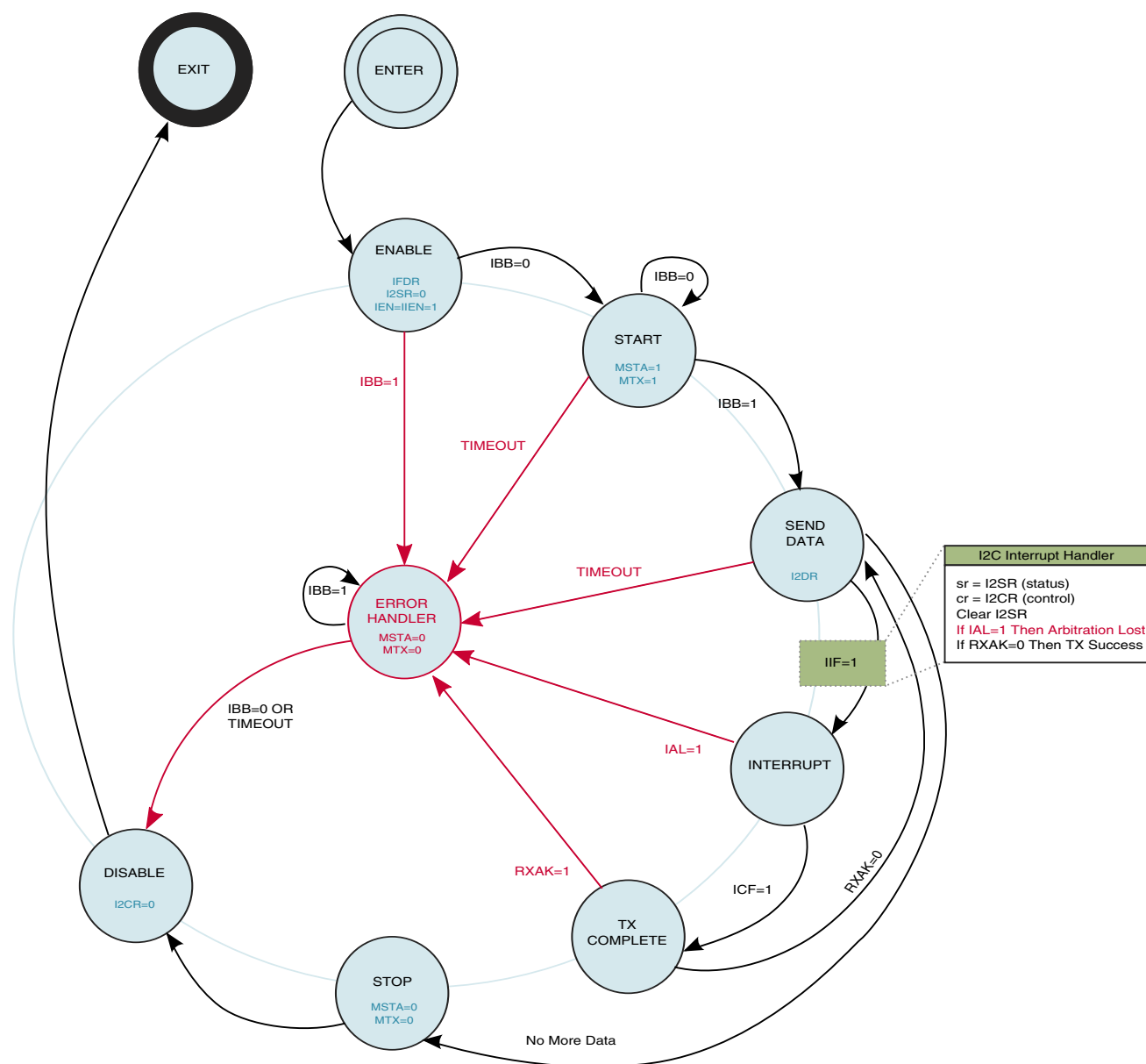


Figure 29-4. I2C Programming state diagram

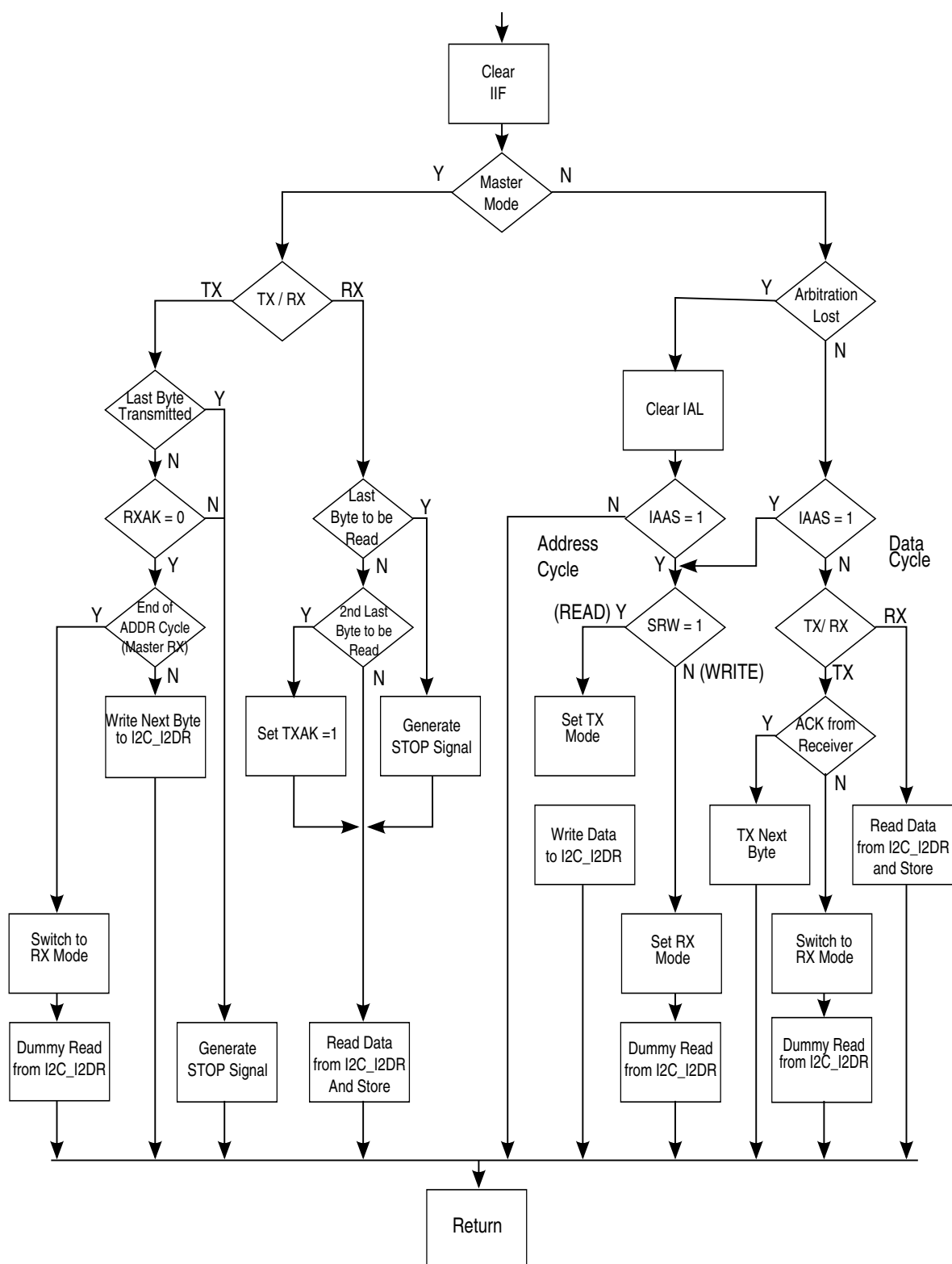


Figure 29-5. Flowchart of typical I2C interrupt routine

**NOTE**

For a Repeated Start only, the Stop-generation stage does not occur in Master mode. A loop repeats itself without stopping for the next start.

For Master Receive mode, I2C is programmed as Master Transmit during Address mode and after slave address transfer; the MTX bit should be cleared and a dummy read on the I2C\_I2DR register should be performed so I2C can read the next receive data.



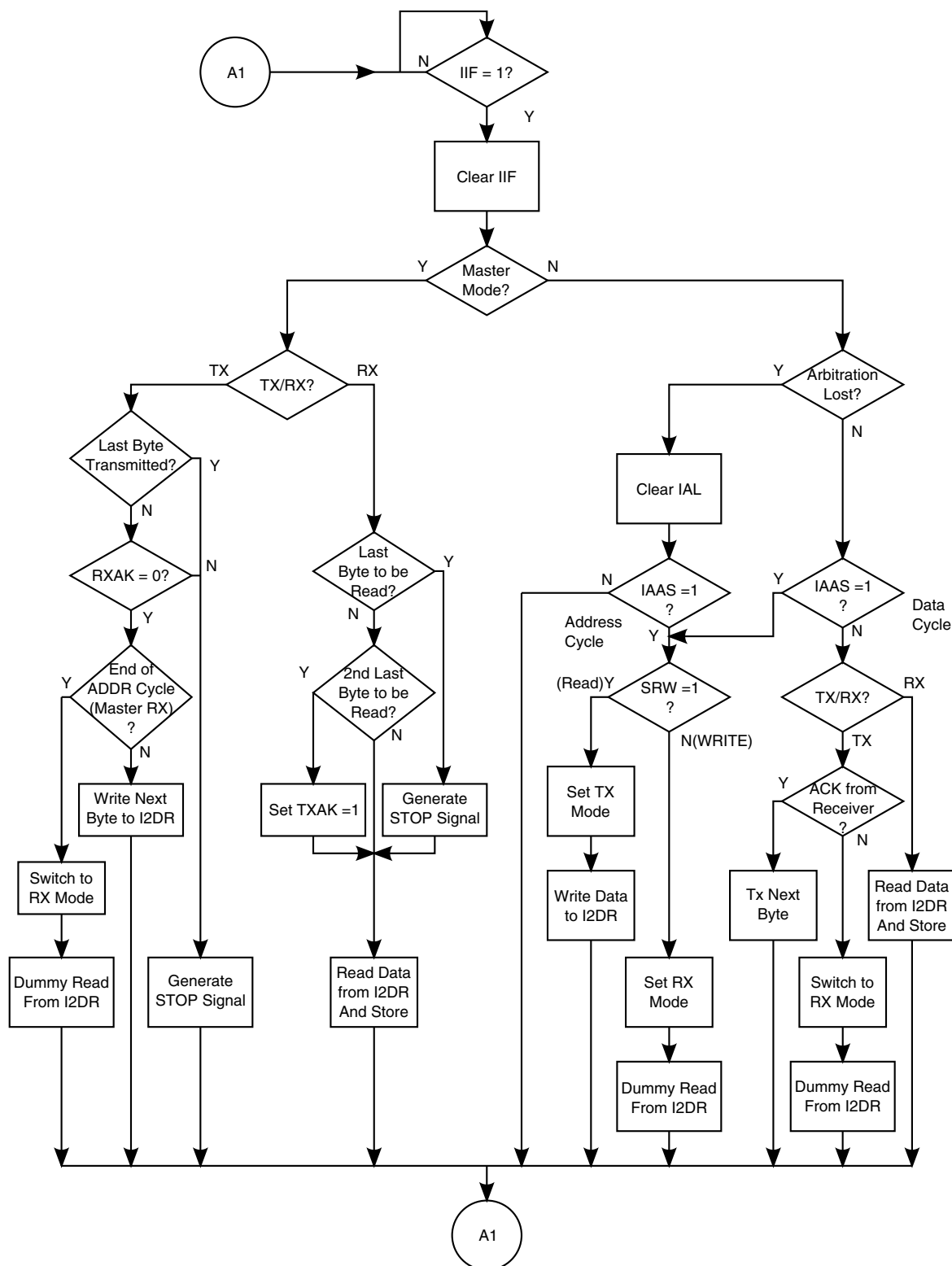
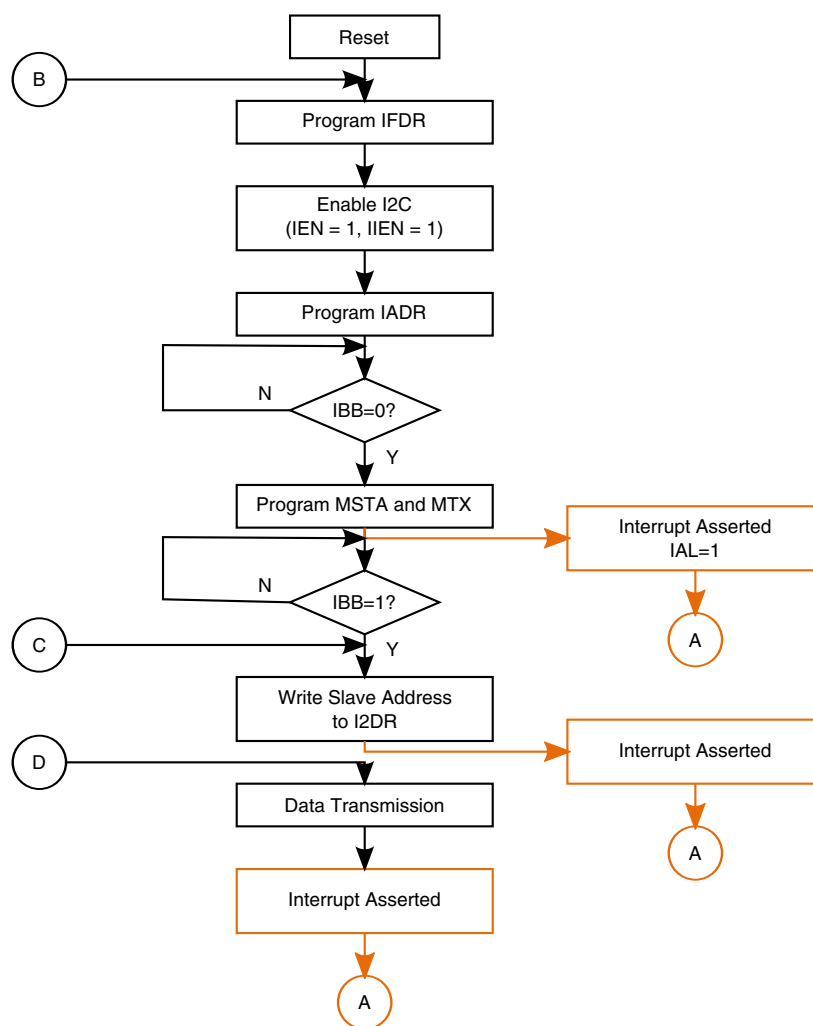


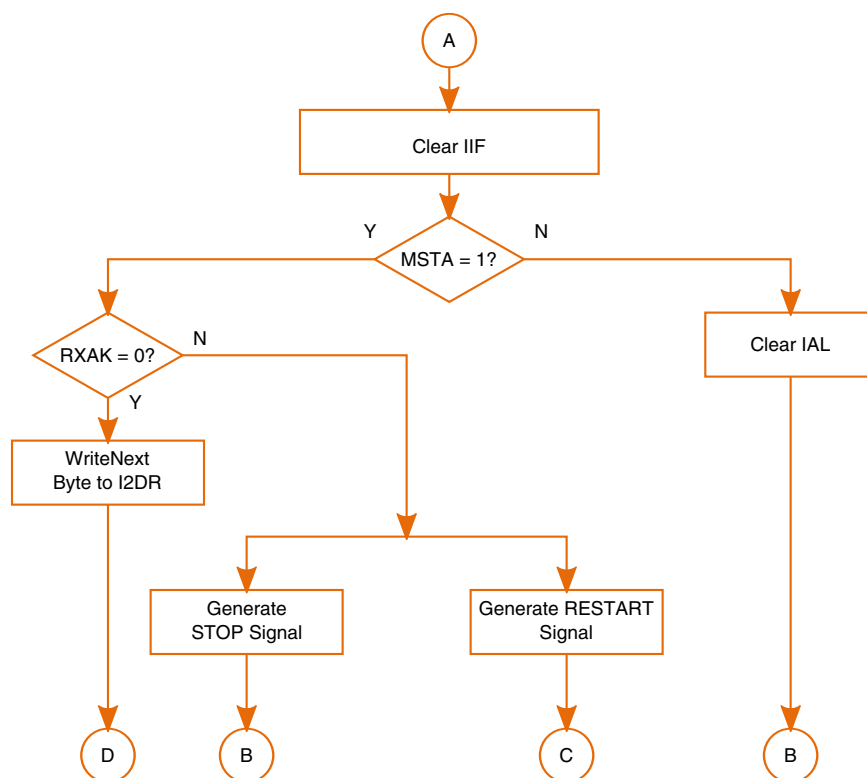
Figure 29-6. Flowchart for typical I2C polling routine

**NOTE**

The timeout value depends on the bus frequency at which I2C is operating. The minimum timeout for polling the IIF bit at a maximum I2C bus frequency of 400 kHz is  $T_{\min} = 25 \mu\text{s}$  ( $=2.5 \times 10 \mu\text{s}$ ). This value can be calculated for any bus frequency. The formula is  $T_{\min} = 10/F_{\text{SCL}}$ , where  $F_{\text{SCL}}$  is the frequency of the I2C clock (SCL).



**Figure 29-7. Detailed flowchart of a typical I2C Master Transmit mode, part 1**



**Figure 29-8. Detailed flowchart of a typical I2C Master Transmit mode, part 2**

Figure 29-7 and Figure 29-8 show the Master Transmit mode operation with interrupt subroutine. If an interrupt is generated and the MSTA bit is 0, then bus arbitration is lost and IAL is set. Software can clear the IAL bit and reprogram I2C. If the MSTA bit is 1, then it is a transfer-generated interrupt. In this case, software can check the RXAK bit for a data receive acknowledgement by the slave and, accordingly, decide to do one of the following:

- Generate a STOP
- Generate a REPEATED START by writing to the I2C\_I2CR register
- Perform the next data transfer by writing to the I2C\_I2DR register

### NOTE

The IBB bit is asserted by a Start condition on the bus, and it is deasserted by a Stop condition on the bus. Therefore, if arbitration is lost due to an unexpected Stop condition during transfer, then IBB is cleared. If arbitration is lost due to a data mismatch, then it is not cleared. Software should always clear the IEN bit and then set it if arbitration is lost.

## 29.6 Software restriction

Software should ensure that there is a delay of at least two module clock cycles after it sets the I2C\_I2CR[RSTA] bit and before writing to the I2C\_I2DR register. The maximum possible clock period of the module clock is 78 ns.

## 29.7 I2C Memory Map/Register Definition

The I2C contains five 16-bit registers.

### NOTE

Registers at offsets 0x0002, 0x0006, 0x000A, and 0x000E are reserved for future additions.

**I2C memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
21A_0000	I2C Address Register (I2C1_IADR)	16	R/W	0000h	<a href="#">29.7.1/1241</a>
21A_0004	I2C Frequency Divider Register (I2C1_IFDR)	16	R/W	0000h	<a href="#">29.7.2/1241</a>
21A_0008	I2C Control Register (I2C1_I2CR)	16	R/W	0000h	<a href="#">29.7.3/1243</a>
21A_000C	I2C Status Register (I2C1_I2SR)	16	R/W	0081h	<a href="#">29.7.4/1244</a>
21A_0010	I2C Data I/O Register (I2C1_I2DR)	16	R/W	0000h	<a href="#">29.7.5/1246</a>
21A_4000	I2C Address Register (I2C2_IADR)	16	R/W	0000h	<a href="#">29.7.1/1241</a>
21A_4004	I2C Frequency Divider Register (I2C2_IFDR)	16	R/W	0000h	<a href="#">29.7.2/1241</a>
21A_4008	I2C Control Register (I2C2_I2CR)	16	R/W	0000h	<a href="#">29.7.3/1243</a>
21A_400C	I2C Status Register (I2C2_I2SR)	16	R/W	0081h	<a href="#">29.7.4/1244</a>
21A_4010	I2C Data I/O Register (I2C2_I2DR)	16	R/W	0000h	<a href="#">29.7.5/1246</a>
21A_8000	I2C Address Register (I2C3_IADR)	16	R/W	0000h	<a href="#">29.7.1/1241</a>
21A_8004	I2C Frequency Divider Register (I2C3_IFDR)	16	R/W	0000h	<a href="#">29.7.2/1241</a>
21A_8008	I2C Control Register (I2C3_I2CR)	16	R/W	0000h	<a href="#">29.7.3/1243</a>
21A_800C	I2C Status Register (I2C3_I2SR)	16	R/W	0081h	<a href="#">29.7.4/1244</a>
21A_8010	I2C Data I/O Register (I2C3_I2DR)	16	R/W	0000h	<a href="#">29.7.5/1246</a>
21F_8000	I2C Address Register (I2C4_IADR)	16	R/W	0000h	<a href="#">29.7.1/1241</a>
21F_8004	I2C Frequency Divider Register (I2C4_IFDR)	16	R/W	0000h	<a href="#">29.7.2/1241</a>
21F_8008	I2C Control Register (I2C4_I2CR)	16	R/W	0000h	<a href="#">29.7.3/1243</a>
21F_800C	I2C Status Register (I2C4_I2SR)	16	R/W	0081h	<a href="#">29.7.4/1244</a>
21F_8010	I2C Data I/O Register (I2C4_I2DR)	16	R/W	0000h	<a href="#">29.7.5/1246</a>

## 29.7.2 I2C Address Register (I2Cx\_IADR)

Address: Base address + 0h offset

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0								ADR							
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**I2Cx\_IADR field descriptions**

Field	Description
15–8 Reserved	This read-only field is reserved and always has the value 0.
7–1 ADR	Slave address. Contains the specific slave address to be used by the I2C. Slave mode is the default I2C mode for an address match on the bus.  <b>NOTE:</b> The I2C_IADR holds the address to which the I2C responds when addressed as a slave. The slave address is not the address sent on the bus during the address transfer. The register is not reset by a software reset.
0 Reserved	This read-only field is reserved and always has the value 0.

## 29.7.3 I2C Frequency Divider Register (I2Cx\_IFDR)

The I2C\_IFDR provides a programmable prescaler to configure the clock for bit-rate selection. The register does not get reset by a software reset.

I2C clock is sourced from PERCLK\_ROOT which is routed from IPG\_CLK\_ROOT. I2C clock frequency can easily be obtained by using the following formula:

**I2C clock Frequency = (PERCLK\_ROOT frequency)/(division factor corresponding to IFDR)**

By default, IPG\_CLK\_ROOT and PERCLK\_ROOT frequencies are set to 49.5MHz, where the root clock is sourced from PLL2's PFD2. Obtaining the frequencies can be accomplished by:

**PLL2 = 528MHz**

**PLL2\_PFD2 = 528MHz \* 18 / 24 = 396MHz**

**IPG\_CLK\_ROOT = (PLL2\_PFD2 / ahb\_podf) / ipg\_podf = (396MHz/4)/2 = 49.5MHz**

**PER\_CLK\_ROOT = IPG\_CLK\_ROOT/perclk\_podf = 49.5MHz/1 = 49.5MHz**

**NOTE**

The above calculation assumes that the default CCM register settings, routing, and division factors are used. If different routing, PFD values, and/or division factors are used, the user must adjust the parameters accordingly to calculate the correct clock frequency.

The following table describes the divider and register values for the register field "IC."

**Table 29-13. I2C\_IFDR Register Field Values**

IC	Divider		IC	Divider		IC	Divider		IC	Divider
0x00	30		0x10	288		0x20	22		0x30	160
0x01	32		0x11	320		0x21	24		0x31	192
0x02	36		0x12	384		0x22	26		0x32	224
0x03	42		0x13	480		0x23	28		0x33	256
0x04	48		0x14	576		0x24	32		0x34	320
0x05	52		0x15	640		0x25	36		0x35	384
0x06	60		0x16	768		0x26	40		0x36	448
0x07	72		0x17	960		0x27	44		0x37	512
0x08	80		0x18	1152		0x28	48		0x38	640
0x09	88		0x19	1280		0x29	56		0x39	768
0x0A	104		0x1A	1536		0x2A	64		0x3A	896
0x0B	128		0x1B	1920		0x2B	72		0x3B	1024
0x0C	144		0x1C	2304		0x2C	80		0x3C	1280
0x0D	160		0x1D	2560		0x2D	96		0x3D	1536
0x0E	192		0x1E	3072		0x2E	112		0x3E	1792
0x0F	240		0x1F	3840		0x2F	128		0x3F	2048

Address: Base address + 4h offset

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0										IC					
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**I2Cx\_IFDR field descriptions**

Field	Description
15–6 Reserved	This read-only field is reserved and always has the value 0.
5–0 IC	I2C clock rate. Prescales the clock for bit-rate selection. Due to potentially slow I2Cn_SCL and I2Cn_SDA rise and fall times, bus signals are sampled at the prescaler frequency. The serial bit clock frequency may be lower than IPG_CLK_ROOT divided by the divider shown in the I2C Data I/O Register.  <b>NOTE:</b> The IC value should not be changed during the data transfer, however, it can be changed before a Repeat Start or Start programming sequence in I2C. The I2C protocol supports bit rates of up to 400 kbps. The IC bits need to be programmed in accordance with this constraint.

## 29.7.4 I2C Control Register (I2Cx\_I2CR)

The I2C\_I2CR is used to enable the I2C and the I2C interrupt. It also contains bits that govern operation as a slave or a master.

Address: Base address + 8h offset

Bit	15	14	13	12	11	10	9	8
Read	0							
Write								
Reset	0	0	0	0	0	0	0	0
Bit	7	6	5	4	3	2	1	0
Read	IEN	IEN	MSTA	MTX	TXAK	0	0	
Write	IEN	IEN	MSTA	MTX	TXAK	RSTA		
Reset	0	0	0	0	0	0	0	0

**I2Cx\_I2CR field descriptions**

Field	Description
15–8 Reserved	This read-only field is reserved and always has the value 0.
7 IEN	<p>I2C enable. Also controls the software reset of the entire I2C. Resetting the bit generates an internal reset to the block. If the block is enabled in the middle of a byte transfer, Slave mode ignores the current bus transfer and starts operating when the next Start condition is detected. Master mode is not aware that the bus is busy, so initiating a start cycle may corrupt the current bus cycle, ultimately causing either the current master or the I2C to lose arbitration. Subsequently, bus operation returns to normal.</p> <p>0 The block is disabled, but registers can still be accessed. 1 The I2C is enabled. This bit must be set before any other I2C_I2CR bits have an effect.</p>
6 I2EN	<p>I2C interrupt enable.</p> <p><b>NOTE:</b> If data is written during the Start condition, that is, just after setting the I2C_I2CR[MSTA] and I2C_I2CR[MTX] bits, then the ICF bit is cleared at the falling edge of SCLK after Start. If data is written after the Start condition and falling edge of SCLK, then the ICF bit is cleared as soon as data is written.</p> <p>0 I2C interrupts are disabled, but the status flag I2C_I2SR[IIF] continues to be set when an Interrupt condition occurs. 1 I2C interrupts are enabled. An I2C interrupt occurs if I2C_I2SR[IIF] is also set.</p>
5 MSTA	<p>Master/Slave mode select bit. If the master loses arbitration, MSTa is cleared without generating a Stop signal.</p> <p><b>NOTE:</b> The module clock should be on for writing to the MSTa bit.</p> <p><b>NOTE:</b> The MSTa bit is cleared by software to generate a Stop condition; it can also be cleared by hardware when the I2C loses the bus arbitration.</p>

*Table continues on the next page...*

**I2Cx\_I2CR field descriptions (continued)**

Field	Description
	0 Slave mode. Changing MSTA from 1 to 0 generates a Stop and selects Slave mode. 1 Master mode. Changing MSTA from 0 to 1 signals a Start on the bus and selects Master mode.
4 MTX	Transmit/Receive mode select bit. Selects the direction of master and slave transfers.  0 Receive. When a slave is addressed, the software should set MTX according to the slave read/write bit in the I2C status register (I2C_I2SR[SRW]). 1 Transmit. In Master mode, MTX should be set according to the type of transfer required. Therefore, for address cycles, MTX is always 1.
3 TXAK	Transmit acknowledge enable. Specifies the value driven onto I2Cn_SDA during acknowledge cycles for both master and slave receivers.  <b>NOTE:</b> Writing TXAK applies only when the I2C bus is a receiver.  0 An acknowledge signal is sent to the bus at the ninth clock bit after receiving one byte of data. 1 No acknowledge signal response is sent (that is, the acknowledge bit = 1).
2 RSTA	Repeat start. Always reads as 0. Attempting a repeat start without bus mastership causes loss of arbitration.  0 No repeat start 1 Generates a Repeated Start condition
1–0 Reserved	This read-only field is reserved and always has the value 0.

**29.7.5 I2C Status Register (I2Cx\_I2SR)**

The I2C\_I2SR contains bits that indicate transaction direction and status.

Address: Base address + Ch offset

Bit	15	14	13	12	11	10	9	8
Read	0							
Write								
Reset	0	0	0	0	0	0	0	0
Bit	7	6	5	4	3	2	1	0
Read	ICF	IAAS	IBB	IAL	0	SRW	IIF	RXAK
Write								
Reset	1	0	0	0	0	0	0	1



## I2Cx\_I2SR field descriptions

Field	Description
15–8 Reserved	This read-only field is reserved and always has the value 0.
7 ICF	Data transferring bit. While one byte of data is transferred, ICF is cleared.  0 Transfer is in progress. 1 Transfer is complete. This bit is set by the falling edge of the ninth clock of the last byte transfer.
6 IAAS	I2C addressed as a slave bit. The ARM platform is interrupted if the interrupt enable (I2C_I2CR[IEN]) is set. The ARM platform must check the slave read/write bit (SRW) and set its Transfer/Receive mode accordingly. Writing to I2C_I2CR clears this bit.  0 Not addressed 1 Addressed as a slave. Set when its own address (I2C_IADR) matches the calling address.
5 IBB	I2C bus busy bit. Indicates the status of the bus.  <b>NOTE:</b> When I2C is enabled (I2C_I2CR[IEN] = 1), it continuously polls the bus data (SDA) and clock (SCL) signals to determine a Start or Stop condition.  0 Bus is idle. If a Stop signal is detected, IBB is cleared. 1 Bus is busy. When Start is detected, IBB is set.
4 IAL	Arbitration lost. Set by hardware in the following circumstances (IAL must be cleared by software by writing a "0" to it at the start of the interrupt service routine): <ul style="list-style-type: none"> <li>I2Cn_SDA input samples low when the master drives high during an address or data-transmit cycle.</li> <li>I2Cn_SDA input samples low when the master drives high during the acknowledge bit of a data-receive cycle.</li> </ul> <p>For the above two cases, the bit is set at the falling edge of the ninth I2Cn_SCL clock during the ACK cycle.</p> <ul style="list-style-type: none"> <li>A Start cycle is attempted when the bus is busy.</li> <li>A Repeated Start cycle is requested in Slave mode.</li> <li>A Stop condition is detected when the master did not request it.</li> </ul> <p><b>NOTE:</b> Software cannot set the bit.</p> 0 No arbitration lost. 1 Arbitration is lost.
3 Reserved	This read-only field is reserved and always has the value 0.
2 SRW	Slave read/write. When the I2C is addressed as a slave, IAAS is set, and the slave read/write bit (SRW) indicates the value of the R/W command bit of the calling address sent from the master. SRW is valid only when a complete transfer has occurred, no other transfers have been initiated, and the I2C is a slave and has an address match.  0 Slave receive, master writing to slave 1 Slave transmit, master reading from slave
1 IIF	I2C interrupt. Must be cleared by the software by writing a "0" to it in the interrupt routine.  <b>NOTE:</b> The software cannot set the bit.  0 No I2C interrupt pending. 1 An interrupt is pending.

*Table continues on the next page...*

**I2Cx\_I2SR field descriptions (continued)**

Field	Description
	<p>This causes a processor interrupt request (if the interrupt enable is asserted [IEN = 1]). The interrupt is set when one of the following occurs:</p> <ul style="list-style-type: none"> <li>• One byte transfer is completed (the interrupt is set at the falling edge of the ninth clock).</li> <li>• An address is received that matches its own specific address in Slave Receive mode.</li> <li>• Arbitration is lost.</li> </ul>
0 RXAK	<p>Received acknowledge. This is the value received from the I2Cn_SDA input for the acknowledge bit during a bus cycle.</p> <p>0 An "acknowledge" signal was received after the completion of an 8-bit data transmission on the bus.  1 A "No acknowledge" signal was detected at the ninth clock.</p>

**29.7.6 I2C Data I/O Register (I2Cx\_I2DR)**

In Master Receive mode, reading the data register allows a read to occur and initiates the next byte to be received. In Slave mode, the same function is available after it is addressed.

Address: Base address + 10h offset

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0								DATA							
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**I2Cx\_I2DR field descriptions**

Field	Description
15–8 Reserved	This read-only field is reserved and always has the value 0.
7–0 DATA	<p>Data Byte. Holds the last data byte received or the next data byte to be transferred. Software writes the next data byte to be transmitted or reads the data byte received.</p> <p><b>NOTE:</b> The core-written value in I2C_I2DR cannot be read back by the core. Only data written by the I2C bus side can be read.</p>

## Chapter 30

# IOMUX Controller (IOMUXC)

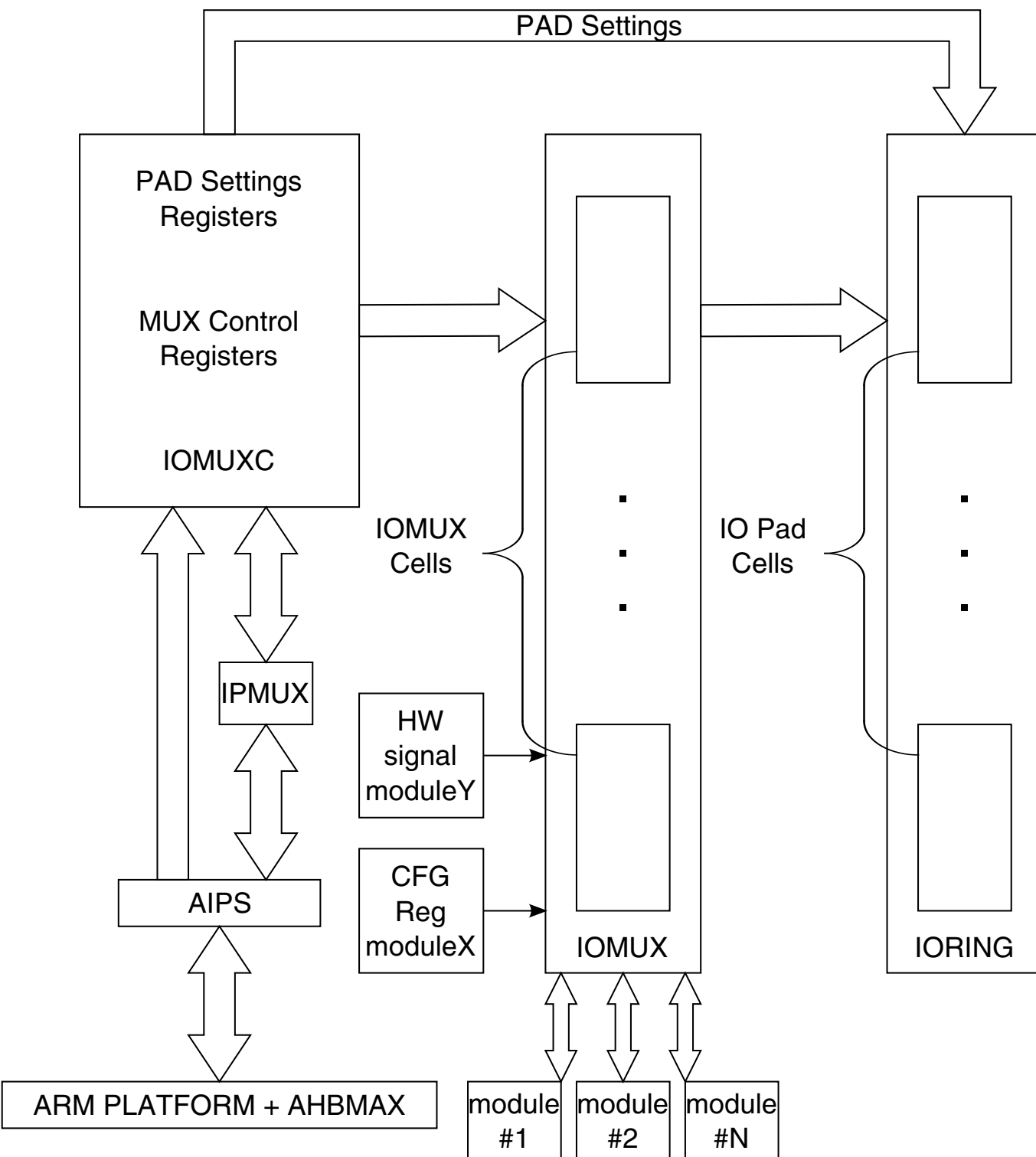
### 30.1 Overview

The IOMUX Controller (IOMUXC), together with the IOMUX, enables the IC to share one pad to several functional blocks. This sharing is done by multiplexing the pad's input and output signals.

Every module requires a specific pad setting (such as pull up or keeper), and for each pad, there are up to 8 muxing options (called ALT modes). The pad settings parameters are controlled by the IOMUXC.

The IOMUX consists only of combinatorial logic combined from several basic IOMUX cells. Each basic IOMUX cell handles only one pad signal's muxing.

[Figure 30-1](#) illustrates the IOMUX/IOMUXC connectivity in the system.



### Figure 30-1. IOMUX SoC Level Block Diagram

### 30.1.1 Features

The IOMUXC features are:

- 32-bit software mux control registers (IOMUXC\_SW\_MUX\_CTL\_PAD\_<PAD NAME> or IOMUXC\_SW\_MUX\_CTL\_GRP\_<GROUP NAME>) to configure 1 of 8 alternate (ALT) MUX\_MODE fields of each pad or a predefined group of pads and to enable the forcing of an input path of the pad(s) (SION bit).
- 32-bit software pad control registers (IOMUXC\_SW\_PAD\_CTL\_PAD\_<PAD NAME> or IOMUXC\_SW\_PAD\_CTL\_GRP\_<GROUP NAME>) to configure specific pad settings of each pad, or a predefined group of pads.
- 32-bit general purpose registers - 14 (GPR0 to GPR13) 32-bit registers according to SoC requirements for any usage.
- 32-bit input select control registers to control the input path to a module when more than one pad drives this module input.

Each SW MUX/PAD CTL IOMUXC register handles only one pad or one pad's group.

Only the minimum number of registers required by software are implemented by hardware. For example, if only ALT0 and ALT1 modes are used on Pad x then only one bit register will be generated as the MUX\_MODE control field in the software mux control register of Pad x.

The software mux control registers may allow the forcing of pads to become input (input path enabled) regardless of the functional direction driven. This may be useful for loopback and GPIO data capture.

## 30.2 Clocks

The table found here describes the clock sources for IOMUXC.

Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 30-1. IOMUXC Clocks**

Clock name	Clock Root	Description
ipt_clk_io	lcdif_pix_clk_root	IO clock
ipg_clk_s	ipg_clk_root	Peripheral access clock

### 30.3 Functional description

This section provides a complete functional description of the block.

The IOMUXC consists of two sub-blocks:

- IOMUXC\_REGISTERS includes all of the IOMUXC registers (see [Features](#)).
- IOMUXC\_LOGIC includes all of the IOMUXC combinatorial logic (IP interface controls, address decoder, observability muxes).

The IOMUX consists of a number (about the number of pads in the SoC) of basic iomux\_cell units. If only one functional mode is required for a specific pad, there is no need for IOMUX and the signals can be connected directly from the module to the I/O. The IOMUX cell is required whenever two or more functional modes are required for a specific pad or when one functional mode and the one test mode are required.

The basic iomux\_cell design, which allows two levels of HW signal control (in ALT6 and ALT7 modes - ALT7 gets highest priority) is shown in [Figure 30-2](#).

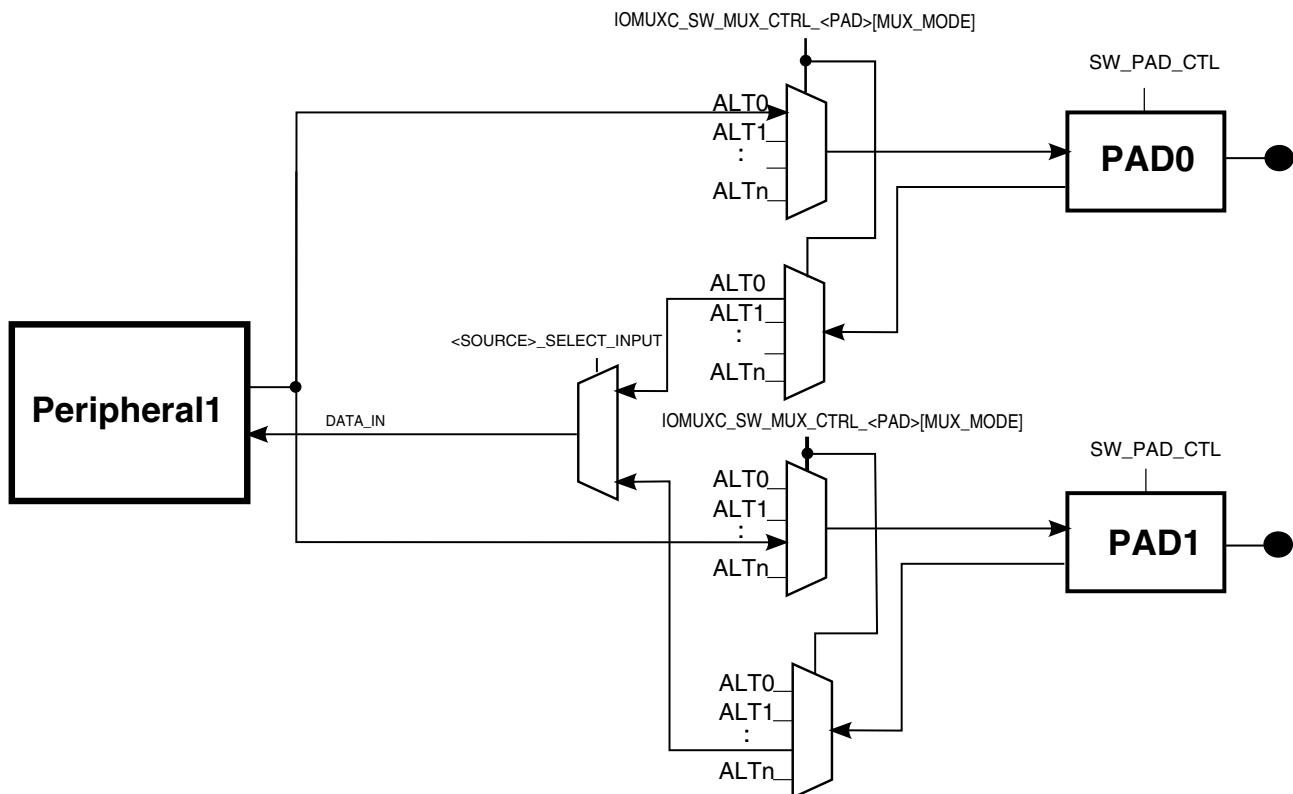


Figure 30-2. IOMUX Cell Block Diagram

### 30.3.1 ALT6 and ALT7 extended muxing modes

The ALT7 and ALT6 extended muxing modes allow any signal in the system (such as fuse, pad input, JTAG, or software register) to override any software configuration and to force the ALT6/ALT7 muxing mode.

It also allows an IOMUX software register to control a group of pads.

### 30.3.2 SW Loopback through SION bit

A limited option exists to override the default pad functionality and force the input path to be active (`ipp_ibe==1'b1`) regardless of the value driven by the corresponding module. This can be done by setting the SION (Software Input On) bit in the IOMUXC\_SW\_MUX\_CTL register (when available) to "1".

Uses include:

- LoopBack - Module x drives the pad and also receives pad value as an input.
- GPIO Capture - Module x drives the pad and the value is captured by GPIO.

### 30.3.3 Daisy chain - multi pads driving same module input pin

In some cases, more than one pad may drive a single module input pin. Such cases require the addition of one more level of IOMUXing; all of these input signals are muxed, and a dedicated software controlled register controls the mux in order to select the required input path.

A block port involved in "daisy chain" requires two software configuration commands, one for selecting the mode for this pad (programmable via the IOMUXC\_SW\_MUX\_CTL\_<PAD> registers) and one for defining it as the input path (via the daisy chain registers).

This means that a block port involved in "daisy chain" requires two software configuration commands, one for selecting the mode for this pad (programmable via the IOMUXC\_SW\_MUX\_CTL\_<PAD> registers) and one for defining it as the input path (via the daisy chain registers). The daisy chain is illustrated in the figure below.

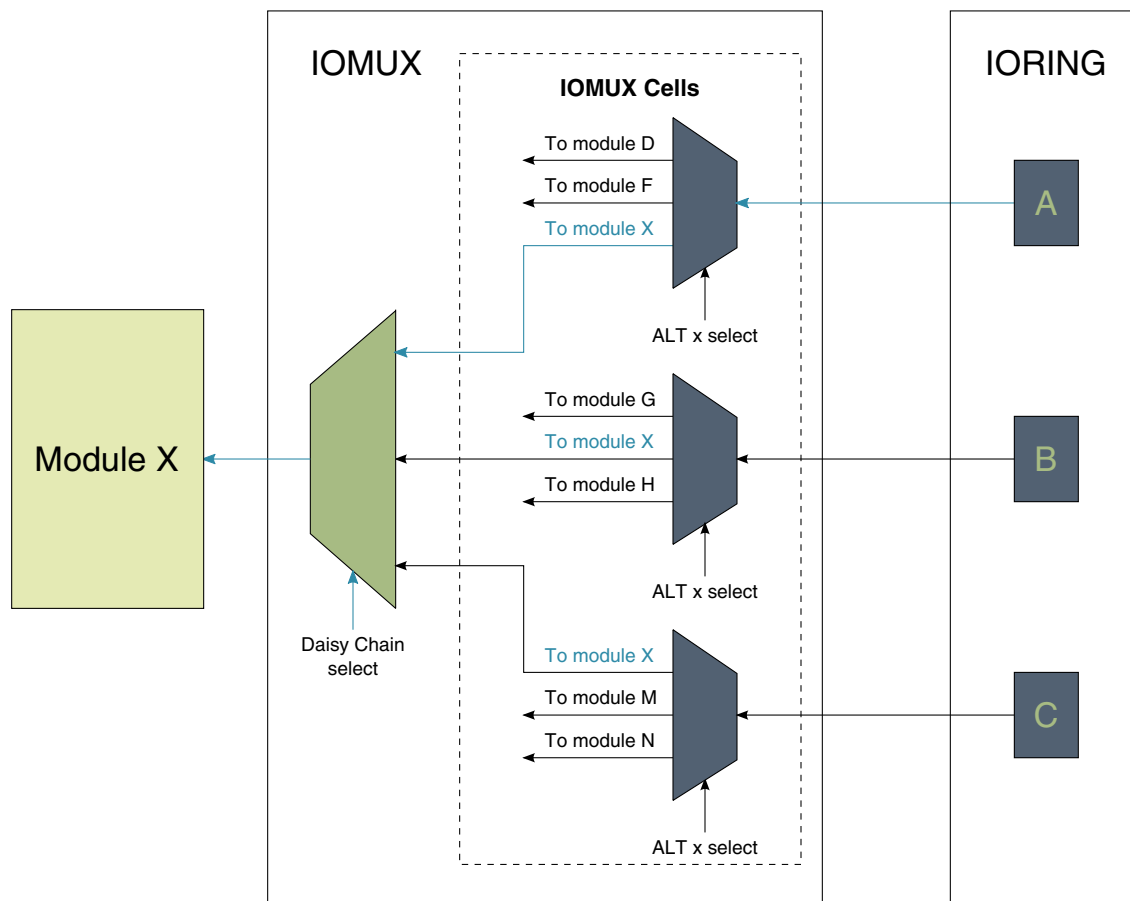


Figure 30-3. Daisy chain illustration



## 30.4 IOMUXC Memory Map/Register Definition

IOMUXC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20E_0000	GPR0 (IOMUXC_GPR0)	32	R/W	0000_0000h	<a href="#">30.4.1/1277</a>
20E_0004	GPR1 (IOMUXC_GPR1)	32	R/W	0040_0005h	<a href="#">30.4.2/1279</a>
20E_0008	GPR2 (IOMUXC_GPR2)	32	R/W	0000_0000h	<a href="#">30.4.3/1281</a>
20E_000C	GPR3 (IOMUXC_GPR3)	32	R/W	01E0_000Fh	<a href="#">30.4.4/1284</a>
20E_0010	GPR4 (IOMUXC_GPR4)	32	R/W	0000_0000h	<a href="#">30.4.5/1288</a>
20E_0014	GPR5 (IOMUXC_GPR5)	32	R/W	0000_0000h	<a href="#">30.4.6/1290</a>
20E_0018	GPR6 (IOMUXC_GPR6)	32	R/W	2222_2222h	<a href="#">30.4.7/1291</a>
20E_001C	GPR7 (IOMUXC_GPR7)	32	R/W	2222_2222h	<a href="#">30.4.8/1291</a>
20E_0020	GPR8 (IOMUXC_GPR8)	32	R/W	0000_0000h	<a href="#">30.4.9/1292</a>
20E_0024	GPR9 (IOMUXC_GPR9)	32	R/W	0000_0000h	<a href="#">30.4.10/1292</a>
20E_0028	GPR10 (IOMUXC_GPR10)	32	R/W	0000_0007h	<a href="#">30.4.11/1293</a>
20E_002C	GPR11 (IOMUXC_GPR11)	32	R/W	0000_0000h	<a href="#">30.4.12/1295</a>
20E_0030	GPR12 (IOMUXC_GPR12)	32	R/W	0F00_0000h	<a href="#">30.4.13/1296</a>
20E_0034	GPR13 (IOMUXC_GPR13)	32	R/W	0000_0008h	<a href="#">30.4.14/1297</a>
20E_004C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_AUD_MCLK)	32	R/W	0000_0005h	<a href="#">30.4.15/1299</a>
20E_0050	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_AUD_RXC)	32	R/W	0000_0005h	<a href="#">30.4.16/1300</a>
20E_0054	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_AUD_RXD)	32	R/W	0000_0005h	<a href="#">30.4.17/1301</a>
20E_0058	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_AUD_RXFS)	32	R/W	0000_0005h	<a href="#">30.4.18/1302</a>
20E_005C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_AUD_TXC)	32	R/W	0000_0005h	<a href="#">30.4.19/1303</a>
20E_0060	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_AUD_TXD)	32	R/W	0000_0005h	<a href="#">30.4.20/1304</a>
20E_0064	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_AUD_TXFS)	32	R/W	0000_0005h	<a href="#">30.4.21/1305</a>
20E_0068	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSP11_MISO)	32	R/W	0000_0005h	<a href="#">30.4.22/1306</a>
20E_006C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSP11_MOSI)	32	R/W	0000_0005h	<a href="#">30.4.23/1307</a>
20E_0070	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSP11_SCLK)	32	R/W	0000_0005h	<a href="#">30.4.24/1308</a>
20E_0074	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSP11_SS0)	32	R/W	0000_0005h	<a href="#">30.4.25/1309</a>

Table continues on the next page...

## IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20E_0078	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPi2_MISO)	32	R/W	0000_0005h	<a href="#">30.4.26/1310</a>
20E_007C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPi2_MOSI)	32	R/W	0000_0005h	<a href="#">30.4.27/1311</a>
20E_0080	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPi2_SCLK)	32	R/W	0000_0005h	<a href="#">30.4.28/1312</a>
20E_0084	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_ECSPi2_SS0)	32	R/W	0000_0005h	<a href="#">30.4.29/1313</a>
20E_0088	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR0)	32	R/W	0000_0005h	<a href="#">30.4.30/1314</a>
20E_008C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_BDR1)	32	R/W	0000_0005h	<a href="#">30.4.31/1315</a>
20E_0090	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA00)	32	R/W	0000_0005h	<a href="#">30.4.32/1316</a>
20E_0094	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA01)	32	R/W	0000_0005h	<a href="#">30.4.33/1317</a>
20E_0098	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA10)	32	R/W	0000_0005h	<a href="#">30.4.34/1318</a>
20E_009C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA11)	32	R/W	0000_0005h	<a href="#">30.4.35/1319</a>
20E_00A0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA12)	32	R/W	0000_0005h	<a href="#">30.4.36/1320</a>
20E_00A4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA13)	32	R/W	0000_0005h	<a href="#">30.4.37/1321</a>
20E_00A8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA14)	32	R/W	0000_0005h	<a href="#">30.4.38/1322</a>
20E_00AC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA15)	32	R/W	0000_0005h	<a href="#">30.4.39/1323</a>
20E_00B0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA02)	32	R/W	0000_0005h	<a href="#">30.4.40/1324</a>
20E_00B4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA03)	32	R/W	0000_0005h	<a href="#">30.4.41/1325</a>
20E_00B8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA04)	32	R/W	0000_0005h	<a href="#">30.4.42/1326</a>
20E_00BC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA05)	32	R/W	0000_0005h	<a href="#">30.4.43/1327</a>
20E_00C0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA06)	32	R/W	0000_0005h	<a href="#">30.4.44/1328</a>
20E_00C4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA07)	32	R/W	0000_0005h	<a href="#">30.4.45/1329</a>
20E_00C8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA08)	32	R/W	0000_0005h	<a href="#">30.4.46/1330</a>
20E_00CC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_DATA09)	32	R/W	0000_0005h	<a href="#">30.4.47/1331</a>

Table continues on the next page...

**IOMUXC memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/ page</b>
20E_00D0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_GDCLK)	32	R/W	0000_0005h	<a href="#">30.4.48/ 1332</a>
20E_00D4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_GDOE)	32	R/W	0000_0005h	<a href="#">30.4.49/ 1333</a>
20E_00D8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_GDRL)	32	R/W	0000_0005h	<a href="#">30.4.50/ 1334</a>
20E_00DC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_GDSP)	32	R/W	0000_0005h	<a href="#">30.4.51/ 1335</a>
20E_00E0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_COM)	32	R/W	0000_0005h	<a href="#">30.4.52/ 1336</a>
20E_00E4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL0)	32	R/W	0000_0005h	<a href="#">30.4.53/ 1337</a>
20E_00E8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL1)	32	R/W	0000_0005h	<a href="#">30.4.54/ 1338</a>
20E_00EC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL2)	32	R/W	0000_0005h	<a href="#">30.4.55/ 1339</a>
20E_00F0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_CTRL3)	32	R/W	0000_0005h	<a href="#">30.4.56/ 1340</a>
20E_00F4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_IRQ)	32	R/W	0000_0005h	<a href="#">30.4.57/ 1341</a>
20E_00F8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_STAT)	32	R/W	0000_0005h	<a href="#">30.4.58/ 1342</a>
20E_00FC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_PWR_WAKE)	32	R/W	0000_0005h	<a href="#">30.4.59/ 1343</a>
20E_0100	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE0)	32	R/W	0000_0005h	<a href="#">30.4.60/ 1344</a>
20E_0104	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE1)	32	R/W	0000_0005h	<a href="#">30.4.61/ 1345</a>
20E_0108	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE2)	32	R/W	0000_0005h	<a href="#">30.4.62/ 1346</a>
20E_010C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCE3)	32	R/W	0000_0005h	<a href="#">30.4.63/ 1347</a>
20E_0110	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_SDCLK)	32	R/W	0000_0005h	<a href="#">30.4.64/ 1348</a>
20E_0114	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_SDLE)	32	R/W	0000_0005h	<a href="#">30.4.65/ 1349</a>
20E_0118	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_SDOE)	32	R/W	0000_0005h	<a href="#">30.4.66/ 1350</a>
20E_011C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_SDSHR)	32	R/W	0000_0005h	<a href="#">30.4.67/ 1351</a>
20E_0120	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM0)	32	R/W	0000_0005h	<a href="#">30.4.68/ 1352</a>
20E_0124	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_EPDC_VCOM1)	32	R/W	0000_0005h	<a href="#">30.4.69/ 1353</a>

*Table continues on the next page...*

## IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20E_0128	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_CRSDV)	32	R/W	0000_0005h	<a href="#">30.4.70/1354</a>
20E_012C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_MDC)	32	R/W	0000_0005h	<a href="#">30.4.71/1355</a>
20E_0130	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_MDIO)	32	R/W	0000_0005h	<a href="#">30.4.72/1356</a>
20E_0134	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_REF_CLK)	32	R/W	0000_0005h	<a href="#">30.4.73/1357</a>
20E_0138	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_RX_ER)	32	R/W	0000_0005h	<a href="#">30.4.74/1358</a>
20E_013C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA0)	32	R/W	0000_0005h	<a href="#">30.4.75/1359</a>
20E_0140	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_RX_DATA1)	32	R/W	0000_0005h	<a href="#">30.4.76/1360</a>
20E_0144	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_TX_CLK)	32	R/W	0000_0005h	<a href="#">30.4.77/1361</a>
20E_0148	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_TX_EN)	32	R/W	0000_0005h	<a href="#">30.4.78/1362</a>
20E_014C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA0)	32	R/W	0000_0005h	<a href="#">30.4.79/1363</a>
20E_0150	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_FEC_TX_DATA1)	32	R/W	0000_0005h	<a href="#">30.4.80/1364</a>
20E_0154	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_USB_H_DATA)	32	R/W	0000_0000h	<a href="#">30.4.81/1365</a>
20E_0158	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_USB_H_STROBE)	32	R/W	0000_0000h	<a href="#">30.4.82/1366</a>
20E_015C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C1_SCL)	32	R/W	0000_0005h	<a href="#">30.4.83/1367</a>
20E_0160	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C1_SDA)	32	R/W	0000_0005h	<a href="#">30.4.84/1368</a>
20E_0164	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C2_SCL)	32	R/W	0000_0005h	<a href="#">30.4.85/1369</a>
20E_0168	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_I2C2_SDA)	32	R/W	0000_0005h	<a href="#">30.4.86/1370</a>
20E_016C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_COL0)	32	R/W	0000_0005h	<a href="#">30.4.87/1371</a>
20E_0170	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_COL1)	32	R/W	0000_0005h	<a href="#">30.4.88/1372</a>
20E_0174	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_COL2)	32	R/W	0000_0005h	<a href="#">30.4.89/1373</a>
20E_0178	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_COL3)	32	R/W	0000_0005h	<a href="#">30.4.90/1374</a>
20E_017C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_COL4)	32	R/W	0000_0005h	<a href="#">30.4.91/1375</a>

Table continues on the next page...

**IOMUXC memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/ page</b>
20E_0180	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_COL5)	32	R/W	0000_0005h	<a href="#">30.4.92/ 1376</a>
20E_0184	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_COL6)	32	R/W	0000_0005h	<a href="#">30.4.93/ 1377</a>
20E_0188	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_COL7)	32	R/W	0000_0005h	<a href="#">30.4.94/ 1378</a>
20E_018C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW0)	32	R/W	0000_0005h	<a href="#">30.4.95/ 1379</a>
20E_0190	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW1)	32	R/W	0000_0005h	<a href="#">30.4.96/ 1380</a>
20E_0194	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW2)	32	R/W	0000_0005h	<a href="#">30.4.97/ 1381</a>
20E_0198	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW3)	32	R/W	0000_0005h	<a href="#">30.4.98/ 1382</a>
20E_019C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW4)	32	R/W	0000_0005h	<a href="#">30.4.99/ 1383</a>
20E_01A0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW5)	32	R/W	0000_0005h	<a href="#">30.4.100/ 1384</a>
20E_01A4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW6)	32	R/W	0000_0005h	<a href="#">30.4.101/ 1385</a>
20E_01A8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_KEY_ROW7)	32	R/W	0000_0005h	<a href="#">30.4.102/ 1386</a>
20E_01AC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_CLK)	32	R/W	0000_0005h	<a href="#">30.4.103/ 1387</a>
20E_01B0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA00)	32	R/W	0000_0005h	<a href="#">30.4.104/ 1388</a>
20E_01B4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA01)	32	R/W	0000_0005h	<a href="#">30.4.105/ 1389</a>
20E_01B8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA10)	32	R/W	0000_0005h	<a href="#">30.4.106/ 1390</a>
20E_01BC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA11)	32	R/W	0000_0005h	<a href="#">30.4.107/ 1391</a>
20E_01C0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA12)	32	R/W	0000_0005h	<a href="#">30.4.108/ 1392</a>
20E_01C4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA13)	32	R/W	0000_0005h	<a href="#">30.4.109/ 1393</a>
20E_01C8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA14)	32	R/W	0000_0005h	<a href="#">30.4.110/ 1394</a>
20E_01CC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA15)	32	R/W	0000_0005h	<a href="#">30.4.111/ 1395</a>
20E_01D0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA16)	32	R/W	0000_0005h	<a href="#">30.4.112/ 1396</a>
20E_01D4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA17)	32	R/W	0000_0005h	<a href="#">30.4.113/ 1397</a>

*Table continues on the next page...*

## IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20E_01D8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA18)	32	R/W	0000_0005h	<a href="#">30.4.114/1398</a>
20E_01DC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA19)	32	R/W	0000_0005h	<a href="#">30.4.115/1399</a>
20E_01E0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA02)	32	R/W	0000_0005h	<a href="#">30.4.116/1400</a>
20E_01E4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA20)	32	R/W	0000_0005h	<a href="#">30.4.117/1401</a>
20E_01E8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA21)	32	R/W	0000_0005h	<a href="#">30.4.118/1402</a>
20E_01EC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA22)	32	R/W	0000_0005h	<a href="#">30.4.119/1403</a>
20E_01F0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA23)	32	R/W	0000_0005h	<a href="#">30.4.120/1404</a>
20E_01F4	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA03)	32	R/W	0000_0005h	<a href="#">30.4.121/1405</a>
20E_01F8	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA04)	32	R/W	0000_0005h	<a href="#">30.4.122/1406</a>
20E_01FC	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA05)	32	R/W	0000_0005h	<a href="#">30.4.123/1407</a>
20E_0200	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA06)	32	R/W	0000_0005h	<a href="#">30.4.124/1408</a>
20E_0204	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA07)	32	R/W	0000_0005h	<a href="#">30.4.125/1409</a>
20E_0208	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA08)	32	R/W	0000_0005h	<a href="#">30.4.126/1410</a>
20E_020C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_DATA09)	32	R/W	0000_0005h	<a href="#">30.4.127/1411</a>
20E_0210	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_ENABLE)	32	R/W	0000_0005h	<a href="#">30.4.128/1412</a>
20E_0214	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_HSYNC)	32	R/W	0000_0005h	<a href="#">30.4.129/1413</a>
20E_0218	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_RESET)	32	R/W	0000_0005h	<a href="#">30.4.130/1414</a>
20E_021C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_LCD_VSYNC)	32	R/W	0000_0005h	<a href="#">30.4.131/1415</a>
20E_0220	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_PWM1)	32	R/W	0000_0005h	<a href="#">30.4.132/1416</a>
20E_0224	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_REF_CLK_24M)	32	R/W	0000_0005h	<a href="#">30.4.133/1417</a>
20E_0228	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_REF_CLK_32K)	32	R/W	0000_0005h	<a href="#">30.4.134/1418</a>
20E_022C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_CLK)	32	R/W	0000_0005h	<a href="#">30.4.135/1419</a>

Table continues on the next page...



**IOMUXC memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/ page</b>
20E_0230	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_CMD)	32	R/W	0000_0005h	<a href="#">30.4.136/1420</a>
20E_0234	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA0)	32	R/W	0000_0005h	<a href="#">30.4.137/1421</a>
20E_0238	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA1)	32	R/W	0000_0005h	<a href="#">30.4.138/1422</a>
20E_023C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA2)	32	R/W	0000_0005h	<a href="#">30.4.139/1423</a>
20E_0240	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA3)	32	R/W	0000_0005h	<a href="#">30.4.140/1424</a>
20E_0244	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA4)	32	R/W	0000_0005h	<a href="#">30.4.141/1425</a>
20E_0248	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA5)	32	R/W	0000_0005h	<a href="#">30.4.142/1426</a>
20E_024C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA6)	32	R/W	0000_0005h	<a href="#">30.4.143/1427</a>
20E_0250	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD1_DATA7)	32	R/W	0000_0005h	<a href="#">30.4.144/1428</a>
20E_0254	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_CLK)	32	R/W	0000_0005h	<a href="#">30.4.145/1429</a>
20E_0258	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_CMD)	32	R/W	0000_0005h	<a href="#">30.4.146/1430</a>
20E_025C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA0)	32	R/W	0000_0005h	<a href="#">30.4.147/1431</a>
20E_0260	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA1)	32	R/W	0000_0005h	<a href="#">30.4.148/1432</a>
20E_0264	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA2)	32	R/W	0000_0005h	<a href="#">30.4.149/1433</a>
20E_0268	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA3)	32	R/W	0000_0005h	<a href="#">30.4.150/1434</a>
20E_026C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA4)	32	R/W	0000_0005h	<a href="#">30.4.151/1435</a>
20E_0270	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA5)	32	R/W	0000_0005h	<a href="#">30.4.152/1436</a>
20E_0274	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA6)	32	R/W	0000_0005h	<a href="#">30.4.153/1437</a>
20E_0278	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_DATA7)	32	R/W	0000_0005h	<a href="#">30.4.154/1438</a>
20E_027C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD2_RESET)	32	R/W	0000_0005h	<a href="#">30.4.155/1439</a>
20E_0280	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD3_CLK)	32	R/W	0000_0005h	<a href="#">30.4.156/1440</a>
20E_0284	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD3_CMD)	32	R/W	0000_0005h	<a href="#">30.4.157/1441</a>

*Table continues on the next page...*

## IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20E_0288	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD3_DATA0)	32	R/W	0000_0005h	<a href="#">30.4.158/1442</a>
20E_028C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD3_DATA1)	32	R/W	0000_0005h	<a href="#">30.4.159/1443</a>
20E_0290	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD3_DATA2)	32	R/W	0000_0005h	<a href="#">30.4.160/1444</a>
20E_0294	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_SD3_DATA3)	32	R/W	0000_0005h	<a href="#">30.4.161/1445</a>
20E_0298	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART1_RXD)	32	R/W	0000_0005h	<a href="#">30.4.162/1446</a>
20E_029C	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_UART1_TXD)	32	R/W	0000_0005h	<a href="#">30.4.163/1447</a>
20E_02A0	Pad Mux Register (IOMUXC_SW_MUX_CTL_PAD_WDOG_B)	32	R/W	0000_0005h	<a href="#">30.4.164/1448</a>
20E_02A4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_AUD_MCLK)	32	R/W	0001_10B0h	<a href="#">30.4.165/1449</a>
20E_02A8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_AUD_RXC)	32	R/W	0001_10B0h	<a href="#">30.4.166/1451</a>
20E_02AC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_AUD_RXD)	32	R/W	0001_10B0h	<a href="#">30.4.167/1453</a>
20E_02B0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_AUD_RXFS)	32	R/W	0001_10B0h	<a href="#">30.4.168/1455</a>
20E_02B4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_AUD_TXC)	32	R/W	0001_10B0h	<a href="#">30.4.169/1457</a>
20E_02B8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_AUD_TXD)	32	R/W	0001_10B0h	<a href="#">30.4.170/1459</a>
20E_02BC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_AUD_TXFS)	32	R/W	0001_10B0h	<a href="#">30.4.171/1461</a>
20E_02C0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR00)	32	R/W	0000_8000h	<a href="#">30.4.172/1463</a>
20E_02C4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR01)	32	R/W	0000_8000h	<a href="#">30.4.173/1465</a>
20E_02C8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR10)	32	R/W	0000_8000h	<a href="#">30.4.174/1467</a>
20E_02CC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR11)	32	R/W	0000_8000h	<a href="#">30.4.175/1469</a>
20E_02D0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR12)	32	R/W	0000_8000h	<a href="#">30.4.176/1471</a>
20E_02D4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR13)	32	R/W	0000_8000h	<a href="#">30.4.177/1473</a>
20E_02D8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR14)	32	R/W	0000_8000h	<a href="#">30.4.178/1475</a>
20E_02DC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR15)	32	R/W	0000_8000h	<a href="#">30.4.179/1477</a>

Table continues on the next page...



**IOMUXC memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/ page</b>
20E_02E0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR02)	32	R/W	0000_8000h	<a href="#">30.4.180/1479</a>
20E_02E4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR03)	32	R/W	0000_8000h	<a href="#">30.4.181/1481</a>
20E_02E8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR04)	32	R/W	0000_8000h	<a href="#">30.4.182/1483</a>
20E_02EC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR05)	32	R/W	0000_8000h	<a href="#">30.4.183/1485</a>
20E_02F0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR06)	32	R/W	0000_8000h	<a href="#">30.4.184/1487</a>
20E_02F4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR07)	32	R/W	0000_8000h	<a href="#">30.4.185/1489</a>
20E_02F8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR08)	32	R/W	0000_8000h	<a href="#">30.4.186/1491</a>
20E_02FC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ADDR09)	32	R/W	0000_8000h	<a href="#">30.4.187/1493</a>
20E_0300	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_CAS_B)	32	R/W	0000_8030h	<a href="#">30.4.188/1495</a>
20E_0304	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_CS0_B)	32	R/W	0000_8000h	<a href="#">30.4.189/1497</a>
20E_0308	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_CS1_B)	32	R/W	0000_8000h	<a href="#">30.4.190/1499</a>
20E_030C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM0)	32	R/W	0000_8030h	<a href="#">30.4.191/1501</a>
20E_0310	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM1)	32	R/W	0000_8030h	<a href="#">30.4.192/1503</a>
20E_0314	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM2)	32	R/W	0000_8030h	<a href="#">30.4.193/1505</a>
20E_0318	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_DQM3)	32	R/W	0000_8030h	<a href="#">30.4.194/1507</a>
20E_031C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_RAS_B)	32	R/W	0000_8030h	<a href="#">30.4.195/1509</a>
20E_0320	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_RESET)	32	R/W	0008_3030h	<a href="#">30.4.196/1511</a>
20E_0324	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDBA0)	32	R/W	0000_8000h	<a href="#">30.4.197/1513</a>
20E_0328	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDBA1)	32	R/W	0000_8000h	<a href="#">30.4.198/1515</a>
20E_032C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDBA2)	32	R/W	0000_B000h	<a href="#">30.4.199/1517</a>
20E_0330	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCKE0)	32	R/W	0000_3000h	<a href="#">30.4.200/1519</a>
20E_0334	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCKE1)	32	R/W	0000_3000h	<a href="#">30.4.201/1521</a>

*Table continues on the next page...*

## IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20E_0338	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDCLK0_P)	32	R/W	0000_8030h	<a href="#">30.4.202/1523</a>
20E_033C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ODT0)	32	R/W	0000_3030h	<a href="#">30.4.203/1525</a>
20E_0340	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_ODT1)	32	R/W	0000_3030h	<a href="#">30.4.204/1527</a>
20E_0344	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS0_P)	32	R/W	0000_2030h	<a href="#">30.4.205/1529</a>
20E_0348	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS1_P)	32	R/W	0000_2030h	<a href="#">30.4.206/1531</a>
20E_034C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS2_P)	32	R/W	0000_2030h	<a href="#">30.4.207/1533</a>
20E_0350	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDQS3_P)	32	R/W	0000_2030h	<a href="#">30.4.208/1535</a>
20E_0354	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_DRAM_SDWE_B)	32	R/W	0000_8000h	<a href="#">30.4.209/1537</a>
20E_0358	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSP11_MISO)	32	R/W	0001_10B0h	<a href="#">30.4.210/1539</a>
20E_035C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSP11_MOSI)	32	R/W	0001_10B0h	<a href="#">30.4.211/1541</a>
20E_0360	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSP11_SCLK)	32	R/W	0001_10B0h	<a href="#">30.4.212/1543</a>
20E_0364	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSP11_SS0)	32	R/W	0001_10B0h	<a href="#">30.4.213/1545</a>
20E_0368	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSP12_MISO)	32	R/W	0001_10B0h	<a href="#">30.4.214/1547</a>
20E_036C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSP12_MOSI)	32	R/W	0001_10B0h	<a href="#">30.4.215/1549</a>
20E_0370	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSP12_SCLK)	32	R/W	0001_10B0h	<a href="#">30.4.216/1551</a>
20E_0374	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_ECSP12_SS0)	32	R/W	0001_10B0h	<a href="#">30.4.217/1553</a>
20E_0378	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_BDR0)	32	R/W	0001_10B0h	<a href="#">30.4.218/1555</a>
20E_037C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_BDR1)	32	R/W	0001_10B0h	<a href="#">30.4.219/1557</a>
20E_0380	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA00)	32	R/W	0001_10B0h	<a href="#">30.4.220/1559</a>
20E_0384	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA01)	32	R/W	0001_10B0h	<a href="#">30.4.221/1561</a>
20E_0388	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA10)	32	R/W	0001_10B0h	<a href="#">30.4.222/1563</a>
20E_038C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA11)	32	R/W	0001_10B0h	<a href="#">30.4.223/1565</a>

Table continues on the next page...

**IOMUXC memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/ page</b>
20E_0390	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA12)	32	R/W	0001_10B0h	<a href="#">30.4.224/1567</a>
20E_0394	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA13)	32	R/W	0001_10B0h	<a href="#">30.4.225/1569</a>
20E_0398	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA14)	32	R/W	0001_10B0h	<a href="#">30.4.226/1571</a>
20E_039C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA15)	32	R/W	0001_10B0h	<a href="#">30.4.227/1573</a>
20E_03A0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA02)	32	R/W	0001_10B0h	<a href="#">30.4.228/1575</a>
20E_03A4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA03)	32	R/W	0001_10B0h	<a href="#">30.4.229/1577</a>
20E_03A8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA04)	32	R/W	0001_10B0h	<a href="#">30.4.230/1579</a>
20E_03AC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA05)	32	R/W	0001_10B0h	<a href="#">30.4.231/1581</a>
20E_03B0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA06)	32	R/W	0001_10B0h	<a href="#">30.4.232/1583</a>
20E_03B4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA07)	32	R/W	0001_10B0h	<a href="#">30.4.233/1585</a>
20E_03B8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA08)	32	R/W	0001_10B0h	<a href="#">30.4.234/1587</a>
20E_03BC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_DATA09)	32	R/W	0001_10B0h	<a href="#">30.4.235/1589</a>
20E_03C0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_GDCLK)	32	R/W	0001_10B0h	<a href="#">30.4.236/1591</a>
20E_03C4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_GDOE)	32	R/W	0001_10B0h	<a href="#">30.4.237/1593</a>
20E_03C8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_GDRL)	32	R/W	0001_10B0h	<a href="#">30.4.238/1595</a>
20E_03CC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_GDSP)	32	R/W	0001_10B0h	<a href="#">30.4.239/1597</a>
20E_03D0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_PWR_COM)	32	R/W	0001_10B0h	<a href="#">30.4.240/1599</a>
20E_03D4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_PWR_CTRL0)	32	R/W	0001_10B0h	<a href="#">30.4.241/1601</a>
20E_03D8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_PWR_CTRL1)	32	R/W	0001_10B0h	<a href="#">30.4.242/1603</a>
20E_03DC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_PWR_CTRL2)	32	R/W	0001_10B0h	<a href="#">30.4.243/1605</a>
20E_03E0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_PWR_CTRL3)	32	R/W	0001_10B0h	<a href="#">30.4.244/1607</a>
20E_03E4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_PWR_IRQ)	32	R/W	0001_10B0h	<a href="#">30.4.245/1609</a>

*Table continues on the next page...*

## IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20E_03E8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_PWR_STAT)	32	R/W	0001_10B0h	<a href="#">30.4.246/1611</a>
20E_03EC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_PWR_WAKE)	32	R/W	0001_10B0h	<a href="#">30.4.247/1613</a>
20E_03F0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_SDCE0)	32	R/W	0001_10B0h	<a href="#">30.4.248/1615</a>
20E_03F4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_SDCE1)	32	R/W	0001_10B0h	<a href="#">30.4.249/1617</a>
20E_03F8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_SDCE2)	32	R/W	0001_10B0h	<a href="#">30.4.250/1619</a>
20E_03FC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_SDCE3)	32	R/W	0001_10B0h	<a href="#">30.4.251/1621</a>
20E_0400	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_SDCLK)	32	R/W	0001_10B0h	<a href="#">30.4.252/1623</a>
20E_0404	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_SDLE)	32	R/W	0001_10B0h	<a href="#">30.4.253/1625</a>
20E_0408	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_SDOE)	32	R/W	0001_10B0h	<a href="#">30.4.254/1627</a>
20E_040C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_SDSHR)	32	R/W	0001_10B0h	<a href="#">30.4.255/1629</a>
20E_0410	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_VCOM0)	32	R/W	0001_10B0h	<a href="#">30.4.256/1631</a>
20E_0414	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_EPDC_VCOM1)	32	R/W	0001_10B0h	<a href="#">30.4.257/1633</a>
20E_0418	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_CRSDV)	32	R/W	0001_10B0h	<a href="#">30.4.258/1635</a>
20E_041C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_MDC)	32	R/W	0001_10B0h	<a href="#">30.4.259/1637</a>
20E_0420	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_MDIO)	32	R/W	0001_10B0h	<a href="#">30.4.260/1639</a>
20E_0424	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_REF_CLK)	32	R/W	0001_10B0h	<a href="#">30.4.261/1641</a>
20E_0428	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_RX_ER)	32	R/W	0001_10B0h	<a href="#">30.4.262/1643</a>
20E_042C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_RX_DATA0)	32	R/W	0001_10B0h	<a href="#">30.4.263/1645</a>
20E_0430	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_RX_DATA1)	32	R/W	0001_10B0h	<a href="#">30.4.264/1647</a>
20E_0434	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_TX_CLK)	32	R/W	0001_10B0h	<a href="#">30.4.265/1649</a>
20E_0438	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_TX_EN)	32	R/W	0001_10B0h	<a href="#">30.4.266/1651</a>
20E_043C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_TX_DATA0)	32	R/W	0001_10B0h	<a href="#">30.4.267/1653</a>

Table continues on the next page...

**IOMUXC memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/page</b>
20E_0440	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_FEC_TX_DATA1)	32	R/W	0001_10B0h	<a href="#">30.4.268/1655</a>
20E_0444	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_USB_H_DATA)	32	R/W	0000_3030h	<a href="#">30.4.269/1657</a>
20E_0448	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_USB_H_STROBE)	32	R/W	0000_3030h	<a href="#">30.4.270/1659</a>
20E_044C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C1_SCL)	32	R/W	0001_10B0h	<a href="#">30.4.271/1661</a>
20E_0450	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C1_SDA)	32	R/W	0001_10B0h	<a href="#">30.4.272/1663</a>
20E_0454	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C2_SCL)	32	R/W	0001_10B0h	<a href="#">30.4.273/1665</a>
20E_0458	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_I2C2_SDA)	32	R/W	0001_10B0h	<a href="#">30.4.274/1667</a>
20E_045C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_MOD)	32	R/W	0000_B060h	<a href="#">30.4.275/1669</a>
20E_0460	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TCK)	32	R/W	0000_7060h	<a href="#">30.4.276/1671</a>
20E_0464	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TDI)	32	R/W	0000_7060h	<a href="#">30.4.277/1673</a>
20E_0468	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TDO)	32	R/W	0000_90B1h	<a href="#">30.4.278/1675</a>
20E_046C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TMS)	32	R/W	0000_7060h	<a href="#">30.4.279/1677</a>
20E_0470	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_JTAG_TRSTB)	32	R/W	0000_7060h	<a href="#">30.4.280/1679</a>
20E_0474	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_COL0)	32	R/W	0001_10B0h	<a href="#">30.4.281/1681</a>
20E_0478	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_COL1)	32	R/W	0001_10B0h	<a href="#">30.4.282/1683</a>
20E_047C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_COL2)	32	R/W	0001_10B0h	<a href="#">30.4.283/1685</a>
20E_0480	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_COL3)	32	R/W	0001_10B0h	<a href="#">30.4.284/1687</a>
20E_0484	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_COL4)	32	R/W	0001_10B0h	<a href="#">30.4.285/1689</a>
20E_0488	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_COL5)	32	R/W	0001_10B0h	<a href="#">30.4.286/1691</a>
20E_048C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_COL6)	32	R/W	0001_10B0h	<a href="#">30.4.287/1693</a>
20E_0490	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_COL7)	32	R/W	0001_10B0h	<a href="#">30.4.288/1695</a>
20E_0494	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW0)	32	R/W	0001_10B0h	<a href="#">30.4.289/1697</a>

*Table continues on the next page...*

## IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20E_0498	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW1)	32	R/W	0001_10B0h	<a href="#">30.4.290/1699</a>
20E_049C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW2)	32	R/W	0001_10B0h	<a href="#">30.4.291/1701</a>
20E_04A0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW3)	32	R/W	0001_10B0h	<a href="#">30.4.292/1703</a>
20E_04A4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW4)	32	R/W	0001_10B0h	<a href="#">30.4.293/1705</a>
20E_04A8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW5)	32	R/W	0001_10B0h	<a href="#">30.4.294/1707</a>
20E_04AC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW6)	32	R/W	0001_10B0h	<a href="#">30.4.295/1709</a>
20E_04B0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_KEY_ROW7)	32	R/W	0001_10B0h	<a href="#">30.4.296/1711</a>
20E_04B4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_CLK)	32	R/W	0001_10B0h	<a href="#">30.4.297/1713</a>
20E_04B8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA00)	32	R/W	0001_10B0h	<a href="#">30.4.298/1715</a>
20E_04BC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA01)	32	R/W	0001_10B0h	<a href="#">30.4.299/1717</a>
20E_04C0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA10)	32	R/W	0001_10B0h	<a href="#">30.4.300/1719</a>
20E_04C4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA11)	32	R/W	0001_10B0h	<a href="#">30.4.301/1721</a>
20E_04C8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA12)	32	R/W	0001_10B0h	<a href="#">30.4.302/1723</a>
20E_04CC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA13)	32	R/W	0001_10B0h	<a href="#">30.4.303/1725</a>
20E_04D0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA14)	32	R/W	0001_10B0h	<a href="#">30.4.304/1727</a>
20E_04D4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA15)	32	R/W	0001_10B0h	<a href="#">30.4.305/1729</a>
20E_04D8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA16)	32	R/W	0001_10B0h	<a href="#">30.4.306/1731</a>
20E_04DC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA17)	32	R/W	0001_10B0h	<a href="#">30.4.307/1733</a>
20E_04E0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA18)	32	R/W	0001_10B0h	<a href="#">30.4.308/1735</a>
20E_04E4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA19)	32	R/W	0001_10B0h	<a href="#">30.4.309/1737</a>
20E_04E8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA02)	32	R/W	0001_10B0h	<a href="#">30.4.310/1739</a>
20E_04EC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA20)	32	R/W	0001_10B0h	<a href="#">30.4.311/1741</a>

Table continues on the next page...



**IOMUXC memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/page</b>
20E_04F0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA21)	32	R/W	0001_10B0h	<a href="#">30.4.312/1743</a>
20E_04F4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA22)	32	R/W	0001_10B0h	<a href="#">30.4.313/1745</a>
20E_04F8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA23)	32	R/W	0001_10B0h	<a href="#">30.4.314/1747</a>
20E_04FC	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA03)	32	R/W	0001_10B0h	<a href="#">30.4.315/1749</a>
20E_0500	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA04)	32	R/W	0001_10B0h	<a href="#">30.4.316/1751</a>
20E_0504	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA05)	32	R/W	0001_10B0h	<a href="#">30.4.317/1753</a>
20E_0508	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA06)	32	R/W	0001_10B0h	<a href="#">30.4.318/1755</a>
20E_050C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA07)	32	R/W	0001_10B0h	<a href="#">30.4.319/1757</a>
20E_0510	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA08)	32	R/W	0001_10B0h	<a href="#">30.4.320/1759</a>
20E_0514	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_DATA09)	32	R/W	0001_10B0h	<a href="#">30.4.321/1761</a>
20E_0518	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_ENABLE)	32	R/W	0001_10B0h	<a href="#">30.4.322/1763</a>
20E_051C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_HSYNC)	32	R/W	0001_10B0h	<a href="#">30.4.323/1765</a>
20E_0520	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_RESET)	32	R/W	0001_10B0h	<a href="#">30.4.324/1767</a>
20E_0524	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_LCD_VSYNC)	32	R/W	0001_10B0h	<a href="#">30.4.325/1769</a>
20E_0528	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_PWM1)	32	R/W	0001_10B0h	<a href="#">30.4.326/1771</a>
20E_052C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_REF_CLK_24M)	32	R/W	0001_10B0h	<a href="#">30.4.327/1773</a>
20E_0530	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_REF_CLK_32K)	32	R/W	0001_10B0h	<a href="#">30.4.328/1775</a>
20E_0534	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_CLK)	32	R/W	0001_10B0h	<a href="#">30.4.329/1777</a>
20E_0538	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_CMD)	32	R/W	0001_10B0h	<a href="#">30.4.330/1779</a>
20E_053C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA0)	32	R/W	0001_10B0h	<a href="#">30.4.331/1781</a>
20E_0540	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA1)	32	R/W	0001_10B0h	<a href="#">30.4.332/1783</a>
20E_0544	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA2)	32	R/W	0001_10B0h	<a href="#">30.4.333/1785</a>

*Table continues on the next page...*

## IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20E_0548	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA3)	32	R/W	0001_10B0h	<a href="#">30.4.334/1787</a>
20E_054C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA4)	32	R/W	0001_10B0h	<a href="#">30.4.335/1789</a>
20E_0550	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA5)	32	R/W	0001_10B0h	<a href="#">30.4.336/1791</a>
20E_0554	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA6)	32	R/W	0001_10B0h	<a href="#">30.4.337/1793</a>
20E_0558	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD1_DATA7)	32	R/W	0001_10B0h	<a href="#">30.4.338/1795</a>
20E_055C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_CLK)	32	R/W	0001_10B0h	<a href="#">30.4.339/1797</a>
20E_0560	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_CMD)	32	R/W	0001_10B0h	<a href="#">30.4.340/1799</a>
20E_0564	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA0)	32	R/W	0001_10B0h	<a href="#">30.4.341/1801</a>
20E_0568	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA1)	32	R/W	0001_10B0h	<a href="#">30.4.342/1803</a>
20E_056C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA2)	32	R/W	0001_10B0h	<a href="#">30.4.343/1805</a>
20E_0570	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA3)	32	R/W	0001_10B0h	<a href="#">30.4.344/1807</a>
20E_0574	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA4)	32	R/W	0001_10B0h	<a href="#">30.4.345/1809</a>
20E_0578	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA5)	32	R/W	0001_10B0h	<a href="#">30.4.346/1811</a>
20E_057C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA6)	32	R/W	0001_10B0h	<a href="#">30.4.347/1813</a>
20E_0580	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_DATA7)	32	R/W	0001_10B0h	<a href="#">30.4.348/1815</a>
20E_0584	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD2_RESET)	32	R/W	0001_10B0h	<a href="#">30.4.349/1817</a>
20E_0588	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD3_CLK)	32	R/W	0001_10B0h	<a href="#">30.4.350/1819</a>
20E_058C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD3_CMD)	32	R/W	0001_10B0h	<a href="#">30.4.351/1821</a>
20E_0590	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD3_DATA0)	32	R/W	0001_10B0h	<a href="#">30.4.352/1823</a>
20E_0594	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD3_DATA1)	32	R/W	0001_10B0h	<a href="#">30.4.353/1825</a>
20E_0598	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD3_DATA2)	32	R/W	0001_10B0h	<a href="#">30.4.354/1827</a>
20E_059C	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_SD3_DATA3)	32	R/W	0001_10B0h	<a href="#">30.4.355/1829</a>

Table continues on the next page...



**IOMUXC memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/page</b>
20E_05A0	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART1_RXD)	32	R/W	0001_10B0h	<a href="#">30.4.356/1831</a>
20E_05A4	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_UART1_TXD)	32	R/W	0001_10B0h	<a href="#">30.4.357/1833</a>
20E_05A8	Pad Control Register (IOMUXC_SW_PAD_CTL_PAD_WDOG_B)	32	R/W	0001_10B0h	<a href="#">30.4.358/1835</a>
20E_05AC	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_ADDDS)	32	R/W	0000_0030h	<a href="#">30.4.359/1836</a>
20E_05B0	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_DDRMODE_CTL)	32	R/W	0000_0000h	<a href="#">30.4.360/1837</a>
20E_05B4	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_DDRPKE)	32	R/W	0000_1000h	<a href="#">30.4.361/1838</a>
20E_05B8	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_DDRPK)	32	R/W	0000_2000h	<a href="#">30.4.362/1839</a>
20E_05BC	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_DDRHYS)	32	R/W	0000_0000h	<a href="#">30.4.363/1840</a>
20E_05C0	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_DDRMODE)	32	R/W	0000_0000h	<a href="#">30.4.364/1841</a>
20E_05C4	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_B0DS)	32	R/W	0000_0030h	<a href="#">30.4.365/1842</a>
20E_05C8	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_CTLDS)	32	R/W	0000_0030h	<a href="#">30.4.366/1842</a>
20E_05CC	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_B1DS)	32	R/W	0000_0030h	<a href="#">30.4.367/1843</a>
20E_05D0	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE)	32	R/W	0008_0000h	<a href="#">30.4.368/1844</a>
20E_05D4	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_B2DS)	32	R/W	0000_0030h	<a href="#">30.4.369/1845</a>
20E_05D8	Pad Group Control Register (IOMUXC_SW_PAD_CTL_GRP_B3DS)	32	R/W	0000_0030h	<a href="#">30.4.370/1845</a>
20E_05DC	Select Input Register (IOMUXC_ANALOG_USB_OTG_ID_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.371/1846</a>
20E_05E0	Select Input Register (IOMUXC_ANALOG_USB_H1_ID_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.372/1847</a>
20E_05E4	Select Input Register (IOMUXC_AUD4_INPUT_DA_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.373/1847</a>
20E_05E8	Select Input Register (IOMUXC_AUD4_INPUT_DB_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.374/1848</a>
20E_05EC	Select Input Register (IOMUXC_AUD4_INPUT_RXCLK_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.375/1849</a>
20E_05F0	Select Input Register (IOMUXC_AUD4_INPUT_RXFS_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.376/1849</a>
20E_05F4	Select Input Register (IOMUXC_AUD4_INPUT_TXCLK_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.377/1850</a>

*Table continues on the next page...*

## IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20E_05F8	Select Input Register (IOMUXC_AUD4_INPUT_TXFS_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.378/1851</a>
20E_05FC	Select Input Register (IOMUXC_AUD5_INPUT_DA_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.379/1851</a>
20E_0600	Select Input Register (IOMUXC_AUD5_INPUT_DB_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.380/1852</a>
20E_0604	Select Input Register (IOMUXC_AUD5_INPUT_RXCLK_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.381/1853</a>
20E_0608	Select Input Register (IOMUXC_AUD5_INPUT_RXFS_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.382/1854</a>
20E_060C	Select Input Register (IOMUXC_AUD5_INPUT_TXCLK_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.383/1855</a>
20E_0610	Select Input Register (IOMUXC_AUD5_INPUT_TXFS_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.384/1856</a>
20E_0614	Select Input Register (IOMUXC_AUD6_INPUT_DA_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.385/1857</a>
20E_0618	Select Input Register (IOMUXC_AUD6_INPUT_DB_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.386/1858</a>
20E_061C	Select Input Register (IOMUXC_AUD6_INPUT_RXCLK_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.387/1859</a>
20E_0620	Select Input Register (IOMUXC_AUD6_INPUT_RXFS_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.388/1860</a>
20E_0624	Select Input Register (IOMUXC_AUD6_INPUT_TXCLK_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.389/1861</a>
20E_0628	Select Input Register (IOMUXC_AUD6_INPUT_TXFS_AMX_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.390/1862</a>
20E_062C	Select Input Register (IOMUXC_CCM_PMIC_READY_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.391/1862</a>
20E_0630	Select Input Register (IOMUXC_CSI_CSI_DATA00_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.392/1863</a>
20E_0634	Select Input Register (IOMUXC_CSI_CSI_DATA01_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.393/1864</a>
20E_0638	Select Input Register (IOMUXC_CSI_CSI_DATA02_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.394/1864</a>
20E_063C	Select Input Register (IOMUXC_CSI_CSI_DATA03_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.395/1865</a>
20E_0640	Select Input Register (IOMUXC_CSI_CSI_DATA04_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.396/1866</a>
20E_0644	Select Input Register (IOMUXC_CSI_CSI_DATA05_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.397/1866</a>
20E_0648	Select Input Register (IOMUXC_CSI_CSI_DATA06_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.398/1867</a>
20E_064C	Select Input Register (IOMUXC_CSI_CSI_DATA07_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.399/1868</a>

Table continues on the next page...

**IOMUXC memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/page</b>
20E_0650	Select Input Register (IOMUXC_CSI_CSI_DATA08_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.400/1868</a>
20E_0654	Select Input Register (IOMUXC_CSI_CSI_DATA09_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.401/1869</a>
20E_0658	Select Input Register (IOMUXC_CSI_CSI_DATA10_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.402/1870</a>
20E_065C	Select Input Register (IOMUXC_CSI_CSI_DATA11_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.403/1870</a>
20E_0660	Select Input Register (IOMUXC_CSI_CSI_DATA12_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.404/1871</a>
20E_0664	Select Input Register (IOMUXC_CSI_CSI_DATA13_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.405/1872</a>
20E_0668	Select Input Register (IOMUXC_CSI_CSI_DATA14_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.406/1873</a>
20E_066C	Select Input Register (IOMUXC_CSI_CSI_DATA15_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.407/1874</a>
20E_0670	Select Input Register (IOMUXC_CSI_CSI_HSYNC_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.408/1874</a>
20E_0674	Select Input Register (IOMUXC_CSI_CSI_PIXCLK_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.409/1875</a>
20E_0678	Select Input Register (IOMUXC_CSI_CSI_VSYNC_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.410/1876</a>
20E_067C	Select Input Register (IOMUXC_ECSP11_CSPI_CLK_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.411/1876</a>
20E_0680	Select Input Register (IOMUXC_ECSP11_DATAREADY_B_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.412/1877</a>
20E_0684	Select Input Register (IOMUXC_ECSP11_MISO_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.413/1878</a>
20E_0688	Select Input Register (IOMUXC_ECSP11_MOSI_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.414/1879</a>
20E_068C	Select Input Register (IOMUXC_ECSP11_SS0_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.415/1880</a>
20E_0690	Select Input Register (IOMUXC_ECSP11_SS1_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.416/1881</a>
20E_0694	Select Input Register (IOMUXC_ECSP11_SS2_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.417/1882</a>
20E_0698	Select Input Register (IOMUXC_ECSP11_SS3_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.418/1883</a>
20E_069C	Select Input Register (IOMUXC_ECSP12_CSPI_CLK_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.419/1883</a>
20E_06A0	Select Input Register (IOMUXC_ECSP12_MISO_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.420/1884</a>
20E_06A4	Select Input Register (IOMUXC_ECSP12_MOSI_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.421/1885</a>

*Table continues on the next page...*

## IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20E_06A8	Select Input Register (IOMUXC_ECSPi2_SS0_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.422/1885</a>
20E_06AC	Select Input Register (IOMUXC_ECSPi2_SS1_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.423/1886</a>
20E_06B0	Select Input Register (IOMUXC_ECSPi3_CSPI_CLK_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.424/1887</a>
20E_06B4	Select Input Register (IOMUXC_ECSPi3_DATAREADY_B_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.425/1887</a>
20E_06B8	Select Input Register (IOMUXC_ECSPi3_MISO_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.426/1888</a>
20E_06BC	Select Input Register (IOMUXC_ECSPi3_MOSI_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.427/1889</a>
20E_06C0	Select Input Register (IOMUXC_ECSPi3_SS0_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.428/1889</a>
20E_06C4	Select Input Register (IOMUXC_ECSPi3_SS1_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.429/1890</a>
20E_06C8	Select Input Register (IOMUXC_ECSPi3_SS2_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.430/1891</a>
20E_06CC	Select Input Register (IOMUXC_ECSPi3_SS3_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.431/1892</a>
20E_06D0	Select Input Register (IOMUXC_ECSPi4_CSPI_CLK_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.432/1892</a>
20E_06D4	Select Input Register (IOMUXC_ECSPi4_MISO_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.433/1893</a>
20E_06D8	Select Input Register (IOMUXC_ECSPi4_MOSI_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.434/1894</a>
20E_06DC	Select Input Register (IOMUXC_ECSPi4_SS0_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.435/1894</a>
20E_06E0	Select Input Register (IOMUXC_ECSPi4_SS1_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.436/1895</a>
20E_06E4	Select Input Register (IOMUXC_ECSPi4_SS2_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.437/1896</a>
20E_06E8	Select Input Register (IOMUXC_EPDC_EPDC_PWR_IRQ_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.438/1897</a>
20E_06EC	Select Input Register (IOMUXC_EPDC_EPDC_PWR_STAT_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.439/1898</a>
20E_06F0	Select Input Register (IOMUXC_FEC_FEC_COL_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.440/1898</a>
20E_06F4	Select Input Register (IOMUXC_FEC_FEC_MDI_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.441/1899</a>
20E_06F8	Select Input Register (IOMUXC_FEC_FEC_RX_DATA0_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.442/1900</a>
20E_06FC	Select Input Register (IOMUXC_FEC_FEC_RX_DATA1_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.443/1900</a>

Table continues on the next page...

**IOMUXC memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/page</b>
20E_0700	Select Input Register (IOMUXC_FEC_FEC_RX_CLK_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.444/1901</a>
20E_0704	Select Input Register (IOMUXC_FEC_FEC_RX_DV_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.445/1902</a>
20E_0708	Select Input Register (IOMUXC_FEC_FEC_RX_ER_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.446/1902</a>
20E_070C	Select Input Register (IOMUXC_FEC_FEC_TX_CLK_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.447/1903</a>
20E_0710	Select Input Register (IOMUXC_GPT_CAPIN1_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.448/1904</a>
20E_0714	Select Input Register (IOMUXC_GPT_CAPIN2_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.449/1905</a>
20E_0718	Select Input Register (IOMUXC_GPT_CLKIN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.450/1906</a>
20E_071C	Select Input Register (IOMUXC_I2C1_SCL_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.451/1906</a>
20E_0720	Select Input Register (IOMUXC_I2C1_SDA_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.452/1907</a>
20E_0724	Select Input Register (IOMUXC_I2C2_SCL_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.453/1908</a>
20E_0728	Select Input Register (IOMUXC_I2C2_SDA_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.454/1908</a>
20E_072C	Select Input Register (IOMUXC_I2C3_SCL_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.455/1909</a>
20E_0730	Select Input Register (IOMUXC_I2C3_SDA_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.456/1910</a>
20E_0734	Select Input Register (IOMUXC_KEY_COL0_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.457/1910</a>
20E_0738	Select Input Register (IOMUXC_KEY_COL1_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.458/1911</a>
20E_073C	Select Input Register (IOMUXC_KEY_COL2_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.459/1912</a>
20E_0740	Select Input Register (IOMUXC_KEY_COL3_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.460/1912</a>
20E_0744	Select Input Register (IOMUXC_KEY_COL4_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.461/1913</a>
20E_0748	Select Input Register (IOMUXC_KEY_COL5_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.462/1914</a>
20E_074C	Select Input Register (IOMUXC_KEY_COL6_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.463/1914</a>
20E_0750	Select Input Register (IOMUXC_KEY_COL7_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.464/1915</a>
20E_0754	Select Input Register (IOMUXC_KEY_ROW0_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.465/1916</a>

*Table continues on the next page...*

## IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20E_0758	Select Input Register (IOMUXC_KEY_ROW1_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.466/1916</a>
20E_075C	Select Input Register (IOMUXC_KEY_ROW2_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.467/1917</a>
20E_0760	Select Input Register (IOMUXC_KEY_ROW3_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.468/1918</a>
20E_0764	Select Input Register (IOMUXC_KEY_ROW4_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.469/1918</a>
20E_0768	Select Input Register (IOMUXC_KEY_ROW5_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.470/1919</a>
20E_076C	Select Input Register (IOMUXC_KEY_ROW6_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.471/1920</a>
20E_0770	Select Input Register (IOMUXC_KEY_ROW7_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.472/1920</a>
20E_0774	Select Input Register (IOMUXC_LCD_BUSY_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.473/1921</a>
20E_0778	Select Input Register (IOMUXC_LCD_DATA00_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.474/1922</a>
20E_077C	Select Input Register (IOMUXC_LCD_DATA01_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.475/1923</a>
20E_0780	Select Input Register (IOMUXC_LCD_DATA02_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.476/1924</a>
20E_0784	Select Input Register (IOMUXC_LCD_DATA03_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.477/1925</a>
20E_0788	Select Input Register (IOMUXC_LCD_DATA04_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.478/1926</a>
20E_078C	Select Input Register (IOMUXC_LCD_DATA05_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.479/1927</a>
20E_0790	Select Input Register (IOMUXC_LCD_DATA06_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.480/1928</a>
20E_0794	Select Input Register (IOMUXC_LCD_DATA07_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.481/1929</a>
20E_0798	Select Input Register (IOMUXC_LCD_DATA08_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.482/1930</a>
20E_079C	Select Input Register (IOMUXC_LCD_DATA09_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.483/1931</a>
20E_07A0	Select Input Register (IOMUXC_LCD_DATA10_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.484/1932</a>
20E_07A4	Select Input Register (IOMUXC_LCD_DATA11_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.485/1933</a>
20E_07A8	Select Input Register (IOMUXC_LCD_DATA12_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.486/1934</a>
20E_07AC	Select Input Register (IOMUXC_LCD_DATA13_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.487/1935</a>

Table continues on the next page...



**IOMUXC memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/page</b>
20E_07B0	Select Input Register (IOMUXC_LCD_DATA14_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.488/1936</a>
20E_07B4	Select Input Register (IOMUXC_LCD_DATA15_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.489/1937</a>
20E_07B8	Select Input Register (IOMUXC_LCD_DATA16_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.490/1938</a>
20E_07BC	Select Input Register (IOMUXC_LCD_DATA17_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.491/1939</a>
20E_07C0	Select Input Register (IOMUXC_LCD_DATA18_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.492/1940</a>
20E_07C4	Select Input Register (IOMUXC_LCD_DATA19_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.493/1941</a>
20E_07C8	Select Input Register (IOMUXC_LCD_DATA20_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.494/1942</a>
20E_07CC	Select Input Register (IOMUXC_LCD_DATA21_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.495/1943</a>
20E_07D0	Select Input Register (IOMUXC_LCD_DATA22_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.496/1944</a>
20E_07D4	Select Input Register (IOMUXC_LCD_DATA23_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.497/1945</a>
20E_07F0	Select Input Register (IOMUXC_SPDIF_SPDIF_IN1_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.498/1945</a>
20E_07F4	Select Input Register (IOMUXC_SPDIF_TX_CLK2_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.499/1946</a>
20E_07F8	Select Input Register (IOMUXC_UART1_UART_RTS_B_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.500/1947</a>
20E_07FC	Select Input Register (IOMUXC_UART1_UART_RX_DATA_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.501/1948</a>
20E_0800	Select Input Register (IOMUXC_UART2_UART_RTS_B_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.502/1948</a>
20E_0804	Select Input Register (IOMUXC_UART2_UART_RX_DATA_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.503/1949</a>
20E_0808	Select Input Register (IOMUXC_UART3_UART_RTS_B_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.504/1950</a>
20E_080C	Select Input Register (IOMUXC_UART3_UART_RX_DATA_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.505/1950</a>
20E_0810	Select Input Register (IOMUXC_UART4_UART_RTS_B_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.506/1951</a>
20E_0814	Select Input Register (IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.507/1952</a>
20E_0818	Select Input Register (IOMUXC_UART5_UART_RTS_B_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.508/1952</a>
20E_081C	Select Input Register (IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.509/1953</a>

*Table continues on the next page...*

## IOMUXC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20E_0820	Select Input Register (IOMUXC_USB_OTG2_OC_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.510/1954</a>
20E_0824	Select Input Register (IOMUXC_USB_OTG1_OC_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.511/1954</a>
20E_0828	Select Input Register (IOMUXC_USDHC1_CARD_DET_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.512/1955</a>
20E_082C	Select Input Register (IOMUXC_USDHC1_WP_ON_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.513/1956</a>
20E_0830	Select Input Register (IOMUXC_USDHC2_CARD_DET_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.514/1956</a>
20E_0834	Select Input Register (IOMUXC_USDHC2_WP_ON_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.515/1957</a>
20E_0838	Select Input Register (IOMUXC_USDHC3_CARD_DET_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.516/1958</a>
20E_083C	Select Input Register (IOMUXC_USDHC3_DATA4_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.517/1958</a>
20E_0840	Select Input Register (IOMUXC_USDHC3_DATA5_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.518/1959</a>
20E_0844	Select Input Register (IOMUXC_USDHC3_DATA6_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.519/1960</a>
20E_0848	Select Input Register (IOMUXC_USDHC3_DATA7_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.520/1961</a>
20E_084C	Select Input Register (IOMUXC_USDHC3_WP_ON_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.521/1961</a>
20E_0850	Select Input Register (IOMUXC_USDHC4_CARD_CLK_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.522/1962</a>
20E_0854	Select Input Register (IOMUXC_USDHC4_CARD_DET_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.523/1963</a>
20E_0858	Select Input Register (IOMUXC_USDHC4_CMD_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.524/1963</a>
20E_085C	Select Input Register (IOMUXC_USDHC4_DATA0_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.525/1964</a>
20E_0860	Select Input Register (IOMUXC_USDHC4_DATA1_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.526/1965</a>
20E_0864	Select Input Register (IOMUXC_USDHC4_DATA2_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.527/1965</a>
20E_0868	Select Input Register (IOMUXC_USDHC4_DATA3_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.528/1966</a>
20E_086C	Select Input Register (IOMUXC_USDHC4_DATA4_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.529/1967</a>
20E_0870	Select Input Register (IOMUXC_USDHC4_DATA5_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.530/1967</a>
20E_0874	Select Input Register (IOMUXC_USDHC4_DATA6_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.531/1968</a>

Table continues on the next page...



**IOMUXC memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20E_0878	Select Input Register (IOMUXC_USDHC4_DATA7_IN_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.532/1969</a>
20E_087C	Select Input Register (IOMUXC_USDHC4_WP_ON_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.533/1969</a>
20E_0880	Select Input Register (IOMUXC_EIM_DTACK_B_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.534/1970</a>
20E_0884	Select Input Register (IOMUXC_EIM_WAIT_B_SELECT_INPUT)	32	R/W	0000_0000h	<a href="#">30.4.535/1971</a>

**30.4.1 GPR0 (IOMUXC\_GPR0)**

Address: 20E\_0000h base + 0h offset = 20E\_0000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								DMAREQ_MUX_SEL7	DMAREQ_MUX_SEL6	DMAREQ_MUX_SEL5	DMAREQ_MUX_SEL4	DMAREQ_MUX_SEL3	DMAREQ_MUX_SEL2	DMAREQ_MUX_SEL1	Reserved
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_GPR0 field descriptions**

Field	Description
31–8 Reserved	This field is reserved. Reserved
7 DMAREQ_MUX_SEL7	Selects between two possible sources for SDMA_EVENT[14]:

*Table continues on the next page...*

**IOMUXC\_GPR0 field descriptions (continued)**

Field	Description
	0 spdif.drq0_spdif_b 1 iomux.sdma_ext_events[1] - External DMA Request via DISP0_DAT17 or GPIO_18
6 DMAREQ_MUX_SEL6	Selects between two possible sources for SDMA_EVENT[23]: 0 esai. 1 i2c3.ipi_int_b
5 DMAREQ_MUX_SEL5	Selects between two possible sources for SDMA_EVENT[9]: 0 ecspi4.ipd_req_cspi_rdma_b 1 epit2.ipi_int_epit_oc
4 DMAREQ_MUX_SEL4	Selects between two possible sources for SDMA_EVENT[10]: 0 ecspi4.ipd_req_cspi_tdma_b 1 i2c1.ipi_int_b
3 DMAREQ_MUX_SEL3	Selects between two possible sources for SDMA_EVENT[5]: 0 ecspi2.ipd_req_cspi_rdma_b 1 i2c1.ipi_int_b
2 DMAREQ_MUX_SEL2	Selects between two possible sources for SDMA_EVENT[4]: 0 ecspi1.ipd_req_cspi_tdma_b 1 i2c2.ipi_int_b
1 DMAREQ_MUX_SEL1	Selects between two possible sources for SDMA_EVENT[3]: 0 ecspi1.ipd_req_cspi_rdma_b 1 i2c3.ipi_int_b
0 -	This field is reserved. Reserved

### 30.4.2 GPR1 (IOMUXC\_GPR1)

Address: 20E\_0000h base + 4h offset = 20E\_0004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								EXC_MON		Reserved			ENET_CLK_SEL_FROM_ANALOG_LOOPBACK		ADD_DS
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	USB_EXP_MODE	ENET_CLK_SEL	Reserved	GINT	ADDRS3[10]	ACT_CS3	ADDRS2[10]	ACT_CS2	ADDRS1[10]	ACT_CS1	ADDRS0[10]	ACT_CS0				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

**IOMUXC\_GPR1 field descriptions**

Field	Description
31–23 Reserved	This field is reserved. Reserved
22 EXC_MON	Exclusive monitor response select of illegal command (of lal gaskets, except MMDC)  0 OKEY response 1 SLVError (default)
21–19 Reserved	This field is reserved. Reserved
18–17 ENET_CLK_SEL_FROM_ANALOG_LOOPBACK	When ENET_CLK_SEL(GPR1[14]) is program to 0(means get enet txclk from internal anatop), use these 2 bits to select from 4 options of loopback path.  00 fec_ref_clk 01 pwm1 10 sd2_rst 11 i2c2_sda
16 ADD_DS	set to 0 will make output driver ~10% stronger at highest strength (DSE=111) for use in case if IO buffer operation at WCS and 200 MHz is marginal  0 output driver ~10% stronger 1 output driver is normal
15 USB_EXP_MODE	USB Exposure mode  0 Exposure mode is disabled. 1 Exposure mode is enabled.
14 ENET_CLK_SEL	choose enet reference clk mode

*Table continues on the next page...*

**IOMUXC\_GPR1 field descriptions (continued)**

Field	Description
	<p>1 get enet tx reference clk from pad (external OSC for both external PHY and Internal Controller)</p> <p>0 get enet tx reference clk from internal clock from ANALOG (loopback through pad), this clock also sent out to external PHY. The loopback path from pad have 4 options, and it's further selected through gpr1[18:17]</p>
13 Reserved	<p>This field is reserved.</p> <p>"usb_otg_id" pin iomux select control.</p> <p>(It functions as the 'select input' mux control)</p> <p>0 selects ENET_RX_ER</p> <p>1 selects GPIO_1.</p>
12 GINT	<p>Global interrupt "0" bit (connected to ARM IRQ#0 and GPC)</p> <p>0 Global interrupt request is not asserted</p> <p>1 Global interrupt request is asserted</p>
11–10 ADDRS3[10]	<p>Active Chip Select and Address Space.</p> <p>Each of the ACT_CSx represents one of the four chip selects of the EIM. When ACT_CSx=1'b1, the corresponding chip select is active and has a valid address space according to its address space configuration determined by ADDRSx[10] bits</p> <p>ADDRSx[10] is setting the space for each chip select which is active. The address space of the first active chip select must be the biggest one, the following active chip select address spaces may be equal or lower.</p> <p>Total address space size is 128 MByte.</p> <p>The supported configurations are:</p> <p>CS0(128M), CS1 (0M), CS2 (0M), CS3(0M) [default configuration]</p> <p>CS0(64M), CS1(64M), CS2(0M), CS3(0M)</p> <p>CS0(64M), CS1(32M), CS2(32M), CS3(0M)</p> <p>CS0(32M), CS1(32M), CS2(32M), CS3(32M)</p> <p>Address Space Configuration options (ADDRSx[10]):</p> <p>00 32 MByte</p> <p>01 64 MByte</p> <p>10 128 MByte</p> <p>11 Reserved</p>
9 ACT_CS3	See description for ADDRS3[10]
8–7 ADDRS2[10]	See description for ADDRS3[10]
6 ACT_CS2	See description for ADDRS3[10]
5–4 ADDRS1[10]	See description for ADDRS3[10]
3 ACT_CS1	See description for ADDRS3[10]
2–1 ADDRS0[10]	See description for ADDRS3[10]

*Table continues on the next page...*

**IOMUXC\_GPR1 field descriptions (continued)**

Field	Description
0 ACT_CS0	See description for ADDRS3[10]

**30.4.3 GPR2 (IOMUXC\_GPR2)**

Address: 20E\_0000h base + 8h offset = 20E\_0008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								L2_MEM_	L2_MEM_	L2_MEM_	L2_MEM_EN_	DCP_MEM_	DCP_MEM_	DCP_MEM_	DCP_MEM_EN_
W									LIGHTSLEEP	DEEPSLEEP	SHUTDOWN	POWERSAVING	LIGHTSLEEP	DEEPSLEEP	SHUTDOWN	POWERSAVING
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LCDIF_MEM_	LCDIF_MEM_	LCDIF_MEM_	LCDIF_MEM_EN_	PXP_MEM_	PXP_MEM_	PXP_MEM_	PXP_MEM_EN_	SPDC_MEM_	SPDC_MEM_	SPDC_MEM_	SPDC_MEM_EN_	EPDC_MEM_	EPDC_MEM_	EPDC_MEM_	EPDC_MEM_EN_
W	LIGHTSLEEP	DEEPSLEEP	SHUTDOWN	POWERSAVING	LIGHTSLEEP	DEEPSLEEP	SHUTDOWN	POWERSAVING	LIGHTSLEEP	DEEPSLEEP	SHUTDOWN	POWERSAVING	LIGHTSLEEP	DEEPSLEEP	SHUTDOWN	POWERSAVING
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_GPR2 field descriptions**

Field	Description
31–24 Reserved	This field is reserved. Reserved
23 L2_MEM_	set to bring memory to light sleep state (Low leakage mode, maintain memory contents, no change to memory output)
22 L2_MEM_	control how memory enter Deep Sleep mode (shutdown periphery power, but maintain memory contents, outputs of memory are pulled low)
DEEPSLEEP	0 no force sleep control supported, memory deep sleep mode only entered when whole system in stop mode 1 force memory into deep sleep mode
21 L2_MEM_	set to bring memory to shutdown state (most power saving state, Shut Down periphery and core, no memory retention)
SHUTDOWN	

*Table continues on the next page...*

## IOMUXC\_GPR2 field descriptions (continued)

Field	Description
20 L2_MEM_EN_ POWERSAVING	enable power saving features on L2 memory  0 none memory power saving features enabled, SHUTDOWN/DEEPSLEEP/LIGHTSLEEP will have no effect 1 memory power saving features enabled, set SHUTDOWN/DEEPSLEEP/LIGHTSLEEP(priority high to low) to enable power saving levels
19 DCP_MEM_ LIGHTSLEEP	set to bring memory to light sleep state (Low leakage mode, maintain memory contents, no change to memory output)
18 DCP_MEM_ DEEPSLEEP	control how memory enter Deep Sleep mode (shutdown periphery power, but maintain memory contents, outputs of memory are pulled low)  0 no force sleep control supported, memory deep sleep mode only entered when whole system in stop mode 1 force memory into deep sleep mode
17 DCP_MEM_ SHUTDOWN	set to bring memory to shutdown state (most power saving state, Shut Down periphery and core, no memory retention)
16 DCP_MEM_EN_ POWERSAVING	enable power saving features on DCP memory  0 none memory power saving features enabled, SHUTDOWN/DEEPSLEEP/LIGHTSLEEP will have no effect 1 memory power saving features enabled, set SHUTDOWN/DEEPSLEEP/LIGHTSLEEP(priority high to low) to enable power saving levels
15 LCDIF_MEM_ LIGHTSLEEP	set to bring memory to light sleep state (Low leakage mode, maintain memory contents, no change to memory output)
14 LCDIF_MEM_ DEEPSLEEP	control how memory enter Deep Sleep mode (shutdown periphery power, but maintain memory contents, outputs of memory are pulled low)  0 no force sleep control supported, memory deep sleep mode only entered when whole system in stop mode 1 force memory into deep sleep mode
13 LCDIF_MEM_ SHUTDOWN	set to bring memory to shutdown state (most power saving state, Shut Down periphery and core, no memory retention)
12 LCDIF_MEM_ EN_ POWERSAVING	enable power saving features on LCDIF memory  0 none memory power saving features enabled, SHUTDOWN/DEEPSLEEP/LIGHTSLEEP will have no effect 1 memory power saving features enabled, set SHUTDOWN/DEEPSLEEP/LIGHTSLEEP(priority high to low) to enable power saving levels
11 PXP_MEM_ LIGHTSLEEP	set to bring memory to light sleep state (Low leakage mode, maintain memory contents, no change to memory output)
10 PXP_MEM_ DEEPSLEEP	control how memory enter Deep Sleep mode (shutdown periphery power, but maintain memory contents, outputs of memory are pulled low)

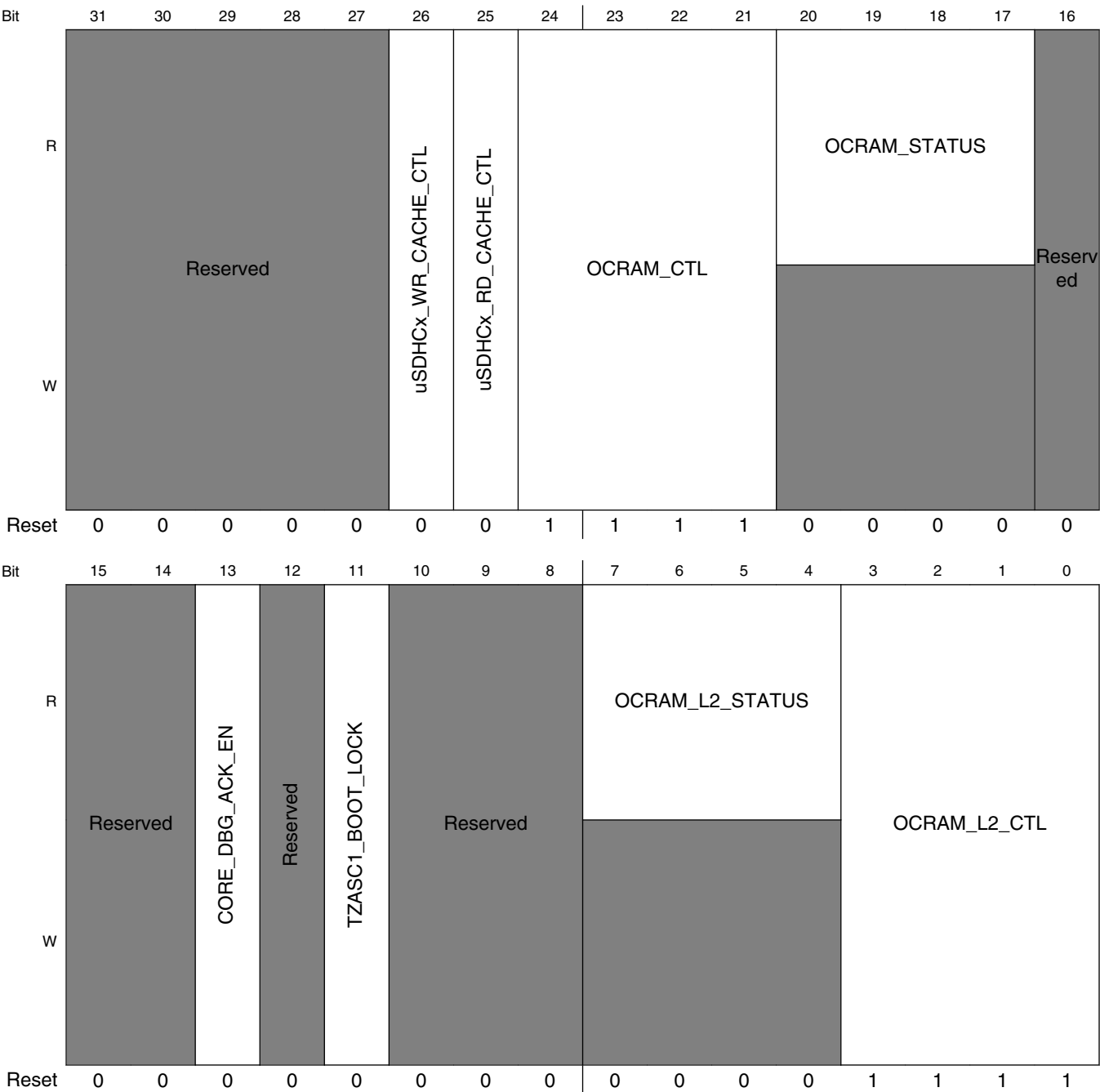
Table continues on the next page...

**IOMUXC\_GPR2 field descriptions (continued)**

<b>Field</b>	<b>Description</b>
	0 no force sleep control supported, memory deep sleep mode only entered when whole system in stop mode 1 force memory into deep sleep mode
9 PXP_MEM_SHUTDOWN	set to bring memory to shutdown state (most power saving state, Shut Down periphery and core, no memory retention)
8 PXP_MEM_EN_POWERSAVING	enable power saving features on PXP memory 0 none memory power saving features enabled, SHUTDOWN/DEEPSLEEP/LIGHTSLEEP will have no effect 1 memory power saving features enabled, set SHUTDOWN/DEEPSLEEP/LIGHTSLEEP(priority high to low) to enable power saving levels
7 SPDC_MEM_LIGHTSLEEP	set to bring memory to light sleep state (Low leakage mode, maintain memory contents, no change to memory output)
6 SPDC_MEM_DEEPSLEEP	control how memory enter Deep Sleep mode (shutdown periphery power, but maintain memory contents, outputs of memory are pulled low) 0 no force sleep control supported, memory deep sleep mode only entered when whole system in stop mode 1 force memory into deep sleep mode
5 SPDC_MEM_SHUTDOWN	set to bring memory to shutdown state (most power saving state, Shut Down periphery and core, no memory retention)
4 SPDC_MEM_EN_POWERSAVING	enable power saving features on SPDC memory 0 none memory power saving features enabled, SHUTDOWN/DEEPSLEEP/LIGHTSLEEP will have no effect 1 memory power saving features enabled, set SHUTDOWN/DEEPSLEEP/LIGHTSLEEP(priority high to low) to enable power saving levels
3 EPDC_MEM_LIGHTSLEEP	set to bring memory to light sleep state (Low leakage mode, maintain memory contents, no change to memory output)
2 EPDC_MEM_DEEPSLEEP	control how memory enter Deep Sleep mode (shutdown periphery power, but maintain memory contents, outputs of memory are pulled low) 0 no force sleep control supported, memory deep sleep mode only entered when whole system in stop mode 1 force memory into deep sleep mode
1 EPDC_MEM_SHUTDOWN	set to bring memory to shutdown state (most power saving state, Shut Down periphery and core, no memory retention)
0 EPDC_MEM_EN_POWERSAVING	enable power saving features on epdc memory 0 none memory power saving features enabled, SHUTDOWN/DEEPSLEEP/LIGHTSLEEP will have no effect 1 memory power saving features enabled, set SHUTDOWN/DEEPSLEEP/LIGHTSLEEP(priority high to low) to enable power saving levels

30.4.4 GPR3 (IOMUXC\_GPR3)

Address: 20E\_0000h base + Ch offset = 20E\_000Ch



IOMUXC\_GPR3 field descriptions

Field	Description
31–27 Reserved	This field is reserved. Reserved

Table continues on the next page...



**IOMUXC\_GPR3 field descriptions (continued)**

Field	Description
26 uSDHCx_WR_CACHE_CTL	Control uSDHCx [1-4] blocks cacheable attribute of AXI write transactions 0 Cacheable attribute is off for write transactions. 1 Cacheable attribute is on for write transactions.
25 uSDHCx_RD_CACHE_CTL	Control uSDHCx [1-4] blocks cacheable attribute of AXI read transactions 0 Cacheable attribute is off for read transactions. 1 Cacheable attribute is on for read transactions.
24–21 OCRAM_CTL	<p>OCRAM_CTL[24] write address pipeline control bit.</p> <p>When this feature is enabled, the write address from the AXI master would be delayed 1 cycle before it can be accepted by the on-chip RAM. This can avoid setup time issue for the write access on the memory cell at high frequency. Enable this feature would cost at most 1 more clock cycle for each AXI write transaction, i.e., at most 1 more clock cycle for each write burst with multiple beats of data. When this feature is disabled, the write address from the AXI master can be accepted by the on-chip RAM without delay, and data can be written to memory at this cycle (if no other access and write data is also ready at this cycle).</p> <p>0 write address pipeline is disabled 1 write address pipeline is enabled</p> <p>OCRAM_CTL[23] - write data pipeline control bit</p> <p>When this feature is enabled, the write data from the AXI master would be delayed 1 cycle before it can be accepted by the on-chip RAM. This can avoid setup time issue for the write access on the memory cell at high frequency. Enable this feature would cost at most 1 more clock cycle for each AXI write transaction, i.e., at most 1 more clock cycle for each write burst with multiple beats of data.</p> <p>When this feature is disabled, the write data from the AXI master can be accepted by the on-chip RAM without delay, and data can be written to memory at this cycle (if no other access and write address is also ready at this cycle).</p> <p>0 write data pipeline is disabled 1 write data pipeline is enabled</p> <p>OCRAM_CTL[22] read address pipeline control bit.</p> <p>When this feature is enabled, the read address from the AXI master would be delayed 1 cycle before it can be accepted by the on-chip RAM. This can avoid setup time issue for the read access on the memory cell at high frequency. Enable this feature would cost at most 1 more clock cycle for each AXI read transaction, i.e., at most 1 more clock cycle for each read burst with multiple beats of data. When this feature is disabled, the read address from the AXI master can be accepted by the on-chip RAM without delay, and data can become ready for master at next clock cycle (if no other access and no read data wait).</p> <p>0 read address pipeline is disabled 1 read address pipeline is enabled</p> <p>OCRAM_CTL[21] - read data wait state control bit</p> <p>When the read data wait state is enabled, it will cost 2 cycles for each read access, (each beat of a read burst). This can avoid the potential timing problem caused by the relatively longer memory access time at higher frequency. When this feature is disabled, it only costs 1 clock cycle to finish a read transaction, i.e., get read data back in the next cycle of read request becomes valid on the bus.</p> <p>0 read data pipeline is disabled 1 read data pipeline is enabled</p>

*Table continues on the next page...*

## IOMUXC\_GPR3 field descriptions (continued)

Field	Description
20–17 OCRAM_STATUS	<p>This field shows the OCRAM pipeline settings status, controlled by OCRAM_CTL[24:21] bits respectively. When the control bit is changed, the corresponding status bit goes high and keeps high until this new configuration is applied the internal logic. This provides a way for software to detect that the configuration has become valid. The suggested flow for changing the configuration in software is:</p> <ul style="list-style-type: none"> <li>• set/clear the control bit</li> <li>• poll the status bit until it goes to 0</li> </ul> <p>OCRAM_STATUS[17] shows the write address pipeline status. This bit value reflects the propagation of the respective control bit to OCRAM memory.</p> <p>OCRAM_STATUS[18] shows the write data pipeline status. This bit value reflects the propagation of the respective control bit to OCRAM memory.</p> <p>OCRAM_STATUS[19] shows the read address pipeline status. This bit value reflects the propagation of the respective control bit to OCRAM memory.</p> <p>OCRAM_STATUS[20] shows the read data pipeline status. This bit value reflects the propagation of the respective control bit to OCRAM memory.</p> <p>0 read data pipeline configuration valid 1 read data pipeline control bit changed</p>
16–14 Reserved	This field is reserved. Reserved
13 CORE_DBG_ACK_EN	Mask control of Core debug acknowledge to global debug acknowledge
	<p>0 Core debug acknowledge is part of global acknowledge.</p> <p>1 Core debug acknowledge is masked by this bit, and it is not part of global acknowledge.</p>
12 Reserved	This field is reserved. Reserved
11 TZASC1_BOOT_LOCK	TZASC-1 secure boot lock
	<p>0 secure boot lock is disabled.</p> <p>1 secure boot lock is enabled</p>
10–8 Reserved	This field is reserved. Reserved
7–4 OCRAM_L2_STATUS	<p>This field shows the OCRAM_L2 pipeline settings status, controlled by OCRAM_L2_CTL[3:0] bits respectively. When the control bit is changed, the corresponding status bit goes high and keeps high until this new configuration is applied the internal logic. This provides a way for software to detect that the configuration has become valid. The suggested flow for changing the configuration in software is:</p> <ul style="list-style-type: none"> <li>• set/clear the control bit</li> <li>• poll the status bit until it goes to 0</li> </ul> <p>OCRAM_L2_STATUS[4] shows the write address pipeline status. This bit value reflects the propagation of the respective control bit to OCRAM_L2 memory.</p> <p>OCRAM_L2_STATUS[5] shows the write data pipeline status. This bit value reflects the propagation of the respective control bit to OCRAM_L2 memory.</p> <p>OCRAM_L2_STATUS[6] shows the read address pipeline status. This bit value reflects the propagation of the respective control bit to OCRAM_L2 memory.</p> <p>OCRAM_L2_STATUS[7] shows the read data pipeline status. This bit value reflects the propagation of the respective control bit to OCRAM_L2 memory.</p>

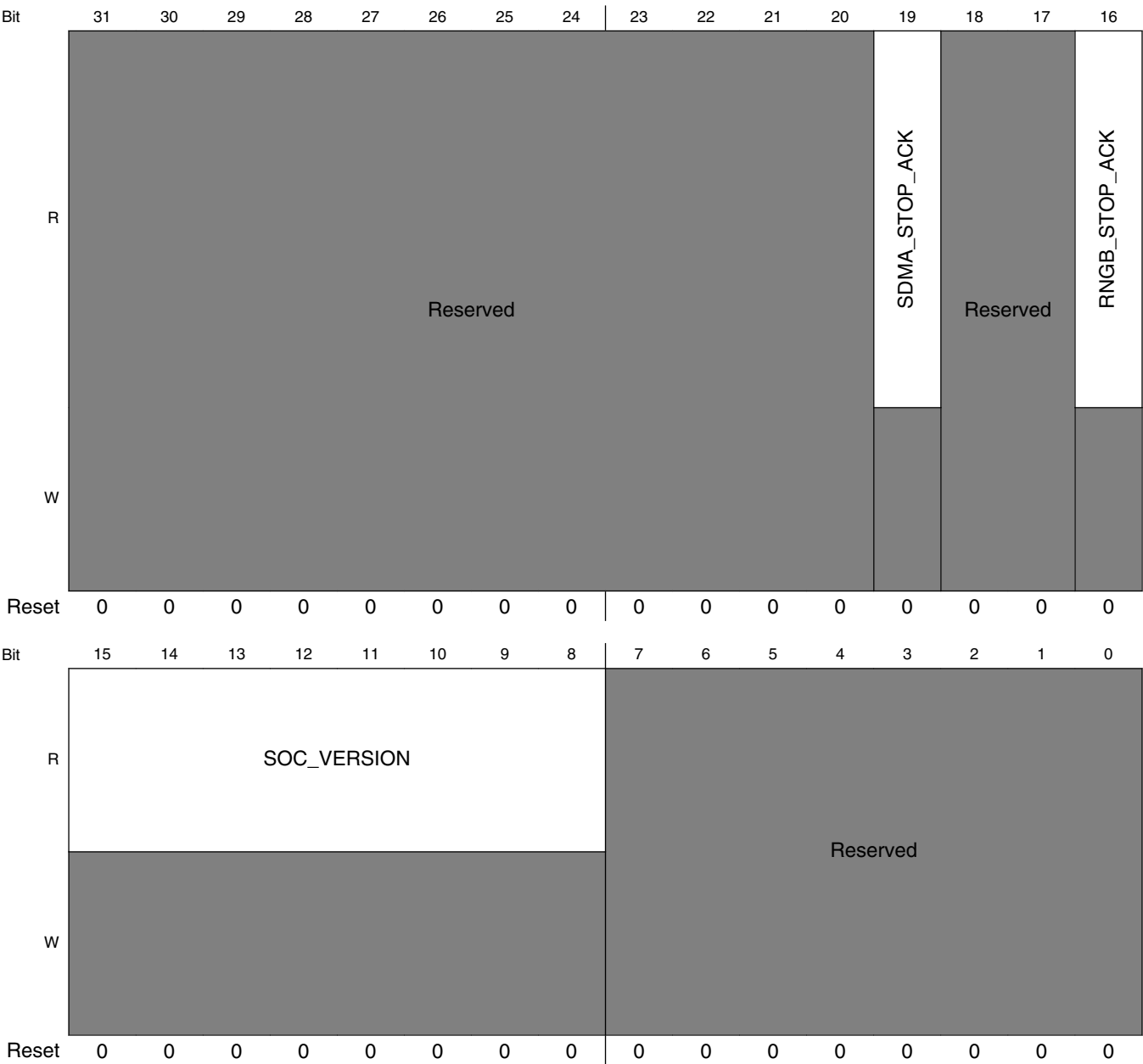
*Table continues on the next page...*

**IOMUXC\_GPR3 field descriptions (continued)**

Field	Description
	<p>0 read data pipeline configuration valid</p> <p>1 read data pipeline control bit changed</p>
3–0 OCRAM_L2_CTL	<p>OCRAM_L2_CTL[3] write address pipeline control bit.</p> <p>When this feature is enabled, the write address from the AXI master would be delayed 1 cycle before it can be accepted by the on-chip RAM. This can avoid setup time issue for the write access on the memory cell at high frequency. Enable this feature would cost at most 1 more clock cycle for each AXI write transaction, i.e., at most 1 more clock cycle for each write burst with multiple beats of data. When this feature is disabled, the write address from the AXI master can be accepted by the on-chip RAM without delay, and data can be written to memory at this cycle (if no other access and write data is also ready at this cycle).</p> <p>0 write address pipeline is disabled</p> <p>1 write address pipeline is enabled</p> <p>OCRAM_L2_CTL[2] - write data pipeline control bit</p> <p>When this feature is enabled, the write data from the AXI master would be delayed 1 cycle before it can be accepted by the on-chip RAM. This can avoid setup time issue for the write access on the memory cell at high frequency. Enable this feature would cost at most 1 more clock cycle for each AXI write transaction, i.e., at most 1 more clock cycle for each write burst with multiple beats of data.</p> <p>When this feature is disabled, the write data from the AXI master can be accepted by the on-chip RAM without delay, and data can be written to memory at this cycle (if no other access and write address is also ready at this cycle).</p> <p>0 write data pipeline is disabled</p> <p>1 write data pipeline is enabled</p> <p>OCRAM_L2_CTL[1] read address pipeline control bit.</p> <p>When this feature is enabled, the read address from the AXI master would be delayed 1 cycle before it can be accepted by the on-chip RAM. This can avoid setup time issue for the read access on the memory cell at high frequency. Enable this feature would cost at most 1 more clock cycle for each AXI read transaction, i.e., at most 1 more clock cycle for each read burst with multiple beats of data. When this feature is disabled, the read address from the AXI master can be accepted by the on-chip RAM without delay, and data can become ready for master at next clock cycle (if no other access and no read data wait).</p> <p>0 read address pipeline is disabled</p> <p>1 read address pipeline is enabled</p> <p>OCRAM_L2_CTL[0] - read data wait state control bit</p> <p>When thread data wait state is enabled, it will cost 2 cycles for each read access, (each beat of a read burst). This can avoid the potential timing problem caused by the relatively longer memory access time at higher frequency. When this feature is disabled, it only costs 1 clock cycle to finish a read transaction, i.e., get read data back in the next cycle of read request becomes valid on the bus.</p> <p>0 read data pipeline is disabled</p> <p>1 read data pipeline is enabled</p>

30.4.5 GPR4 (IOMUXC\_GPR4)

Address: 20E\_0000h base + 10h offset = 20E\_0010h



IOMUXC\_GPR4 field descriptions

Field	Description
31–20 Reserved	This field is reserved. Reserved

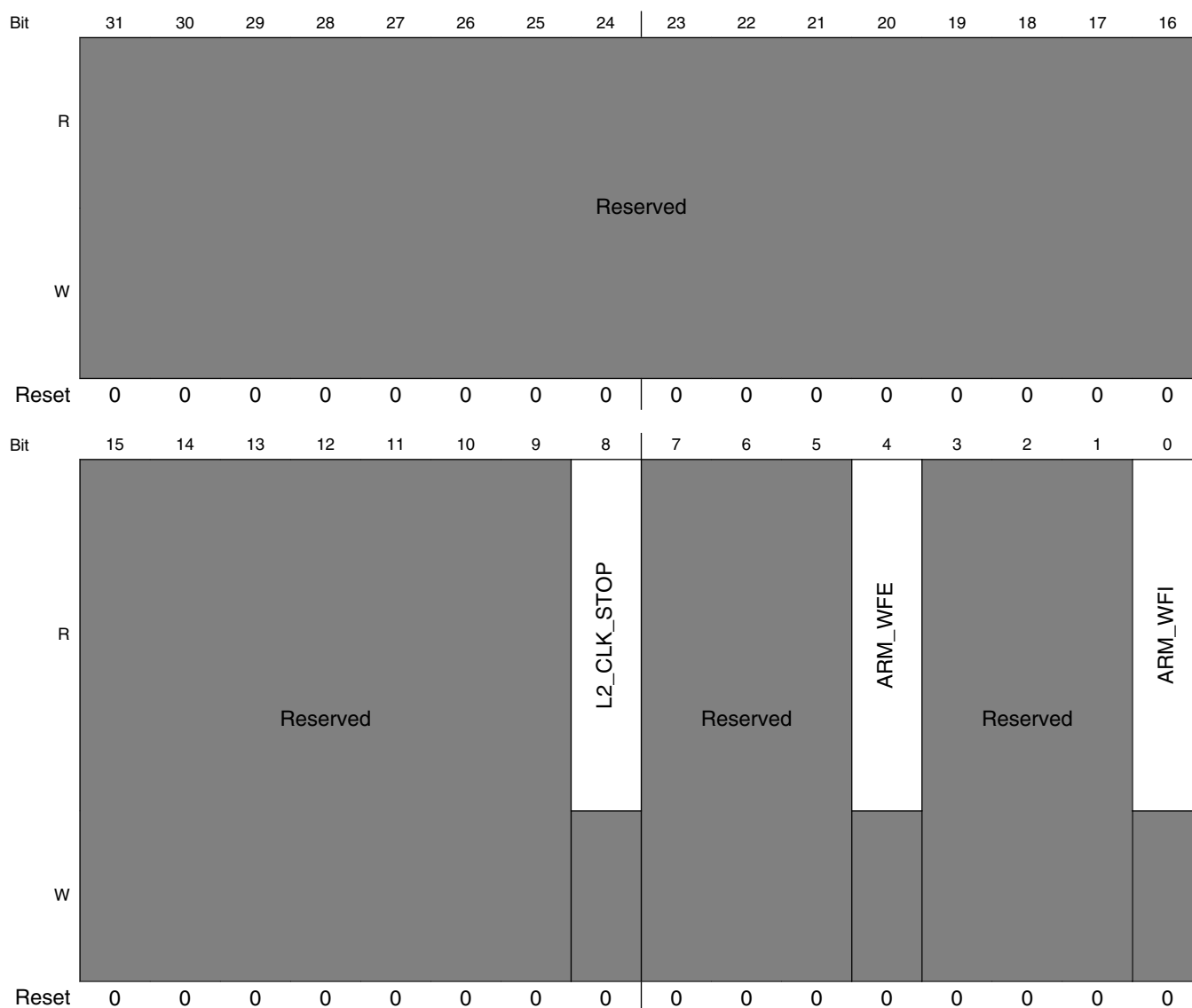
Table continues on the next page...

**IOMUXC\_GPR4 field descriptions (continued)**

<b>Field</b>	<b>Description</b>
19 SDMA_STOP_ ACK	SDMA stop acknowledge. This is status (read only) bit.  0 SDMA stop acknowledge is not asserted. 1 SDMA stop acknowledge is asserted, SDMA is in STOP mode.
18–17 Reserved	This field is reserved. Reserved
16 RNGB_STOP_ ACK	RNGB stop acknowledge. This is status (read only) bit.  0 RNGB stop acknowledge is not asserted. 1 RNGB stop acknowledge is asserted, RNGB is in STOP mode.
15–8 SOC_VERSION	This is status (read only) field.
7–0 Reserved	This field is reserved. Reserved

## 30.4.6 GPR5 (IOMUXC\_GPR5)

Address: 20E\_0000h base + 14h offset = 20E\_0014h











**IOMUXC\_GPR5 field descriptions**

Field	Description
31–9 Reserved	This field is reserved. Reserved
8 L2_CLK_STOP	L2 cache clock stop indication (this is a status, read only bit) 0 L2 cache clock is running 1 L2 cache clock stopped
7–5 Reserved	This field is reserved. Reserved

*Table continues on the next page...*

**IOMUXC\_GPR5 field descriptions (continued)**

Field	Description
4 ARM_WFE	ARM WFE event out indication on WFE state of the cores (these are status, read only bits)  0 ARM Core[GPR5-index - 4] is not in  Wait for Event  mode 1 ARM Core[GPR5-index - 4] is in  Wait for Event  mode
3–1 Reserved	This field is reserved. Reserved
0 ARM_WFI	ARM WFI event out indicating on WFI state of the cores (these are status, read only bits)  0 ARM Core[GPR5-index] is not in  Wait for Interrupt  mode 1 ARM Core[GPR5-index] is in  Wait for Interrupt  mode

**30.4.7 GPR6 (IOMUXC\_GPR6)**

Address: 20E\_0000h base + 18h offset = 20E\_0018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	<div>Reserved</div>																															
W																																
Reset	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0

**IOMUXC\_GPR6 field descriptions**

Field	Description
31–0 Reserved	This field is reserved.

**30.4.8 GPR7 (IOMUXC\_GPR7)**

Address: 20E\_0000h base + 1Ch offset = 20E\_001Ch

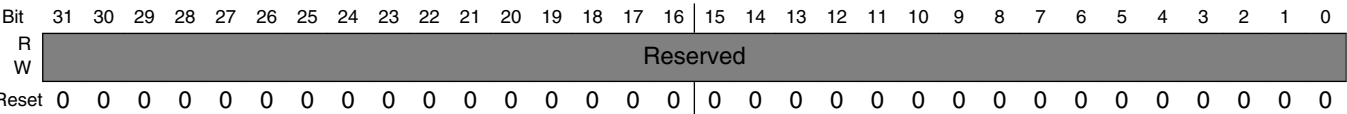
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	<div>Reserved</div>																															
W																																
Reset	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	0

**IOMUXC\_GPR7 field descriptions**

Field	Description
31–0 Reserved	This field is reserved.

30.4.9 GPR8 (IOMUXC\_GPR8)

Address: 20E\_0000h base + 20h offset = 20E\_0020h

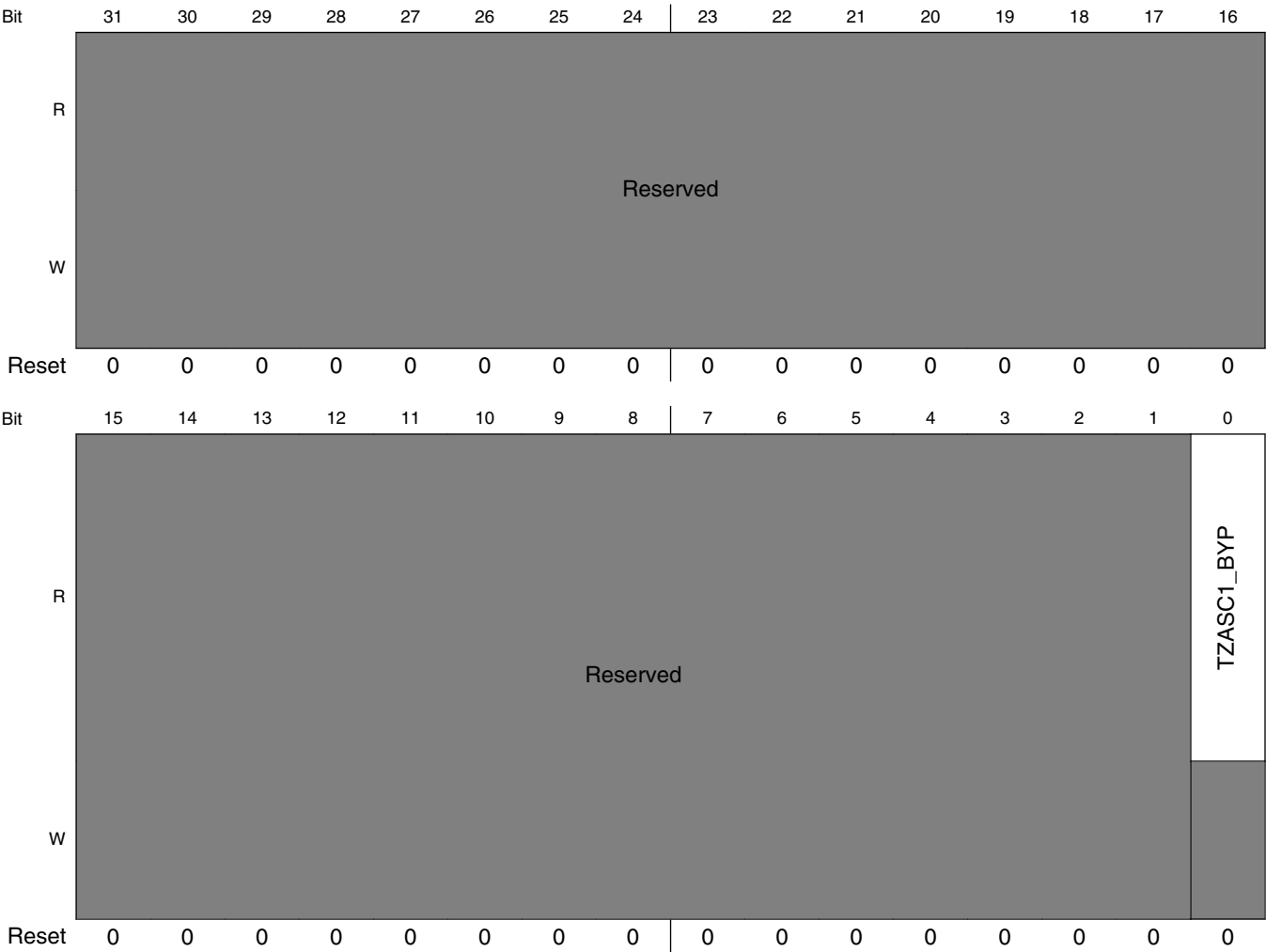


IOMUXC\_GPR8 field descriptions

Field	Description
31–0 Reserved	This field is reserved.

30.4.10 GPR9 (IOMUXC\_GPR9)

Address: 20E\_0000h base + 24h offset = 20E\_0024h





**IOMUXC\_GPR9 field descriptions**

Field	Description
31–1 Reserved	This field is reserved. Reserved
0 TZASC1_BYP	TZASC-1 BYPASS MUX control  0 The TZASC-1 is bypassed and the transactions to DDR are not being checked. 1 The TZASC-1 is not bypassed and the transactions to DDR are being monitored / checked.

**30.4.11 GPR10 (IOMUXC\_GPR10)**

Address: 20E\_0000h base + 28h offset = 20E\_0028h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	LOCK_OCRAM_TZ_ADDR					LOCK_OCRAM_TZ_EN	LOCK_OCRAM_L2_TZ_ADDR							LOCK_OCRAM_L2_TZ_EN	LOCK_SEC_ERR_RESP	LOCK_DBG_CLK_EN	LOCK_DBG_EN
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	OCRAM_TZ_ADDR					OCRAM_TZ_EN	OCRAM_L2_TZ_ADDR							OCRAM_L2_TZ_EN	SEC_ERR_RESP	DBG_CLK_EN	DBG_EN
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	

**IOMUXC\_GPR10 field descriptions**

Field	Description
31–27 LOCK_OCRAM_TZ_ADDR	Lock OCRAM_TZ_ADDR field for changes. This is a sticky field, once set it can't be cleared (only by reset).  0 Field is not locked 1 Field is locked (read access only)
26 LOCK_OCRAM_TZ_EN	Lock OCRAM_TZ_EN field for changes. This is a sticky field, once set it can't be cleared (only by reset).  0 Field is not locked 1 Field is locked (read access only)
25–20 LOCK_OCRAM_L2_TZ_ADDR	Lock OCRAM_L2_TZ_ADDR field for changes. This is a sticky field, once set it can't be cleared (only by reset).

*Table continues on the next page...*

**IOMUXC\_GPR10 field descriptions (continued)**

Field	Description
	0 Field is not locked 1 Field is locked (read access only)
19 LOCK_OCRAM_L2_TZ_EN	Lock OCRAM_L2_TZ_EN field for changes. This is a sticky field, once set it can't be cleared (only by reset). 0 Field is not locked 1 Field is locked (read access only)
18 LOCK_SEC_ERR_RESP	Lock SEC_ERR_RESP field for changes. This is a sticky field, once set it can't be cleared (only by reset). 0 Field is not locked 1 Field is locked (read access only)
17 LOCK_DBG_CLK_EN	Lock DBG_CLK_EN field for changes. This is a sticky field, once set it can't be cleared (only by reset). 0 Field is not locked 1 Field is locked (read access only)
16 LOCK_DBG_EN	Lock DBG_EN field for changes. This is a sticky field, once set it can't be cleared (only by reset). 0 Field is not locked 1 Field is locked (read access only)
15–11 OCRAM_TZ_ADDR	OCRAM TrustZone (TZ) start address. This is the start address of the secure memory region within the OCRAM memory space is 4KB granularity. The start address affects the OCRAM transactions only if OCRAM_TZ_EN bit is set. The OCRAM TZ ENDADDR is not configurable and is set to the end of OCRAM memory space.
10 OCRAM_TZ_EN	OCRAM TrustZone (TZ) enable. 0 The TrustZone feature is disabled. Entire OCRAM space is available for all access types (secure/non-secure/user/supervisor). 1 The TrustZone feature is enabled. Access to address in the range specified by [ENDADDR:STARTADDR] follows the execution mode access policy described in CSU chapter.
9–4 OCRAM_L2_TZ_ADDR	OCRAM_L2 TrustZone (TZ) start address. This is the start address of the secure memory region within the OCRAM_L2 memory space is 4KB granularity. The start address affects the OCRAM_L2 transactions only if OCRAM_L2_TZ_EN bit is set. The OCRAM_L2 TZ ENDADDR is not configurable and is set to the end of OCRAM_L2 memory space.
3 OCRAM_L2_TZ_EN	OCRAM_L2 TrustZone (TZ) enable. 0 The TrustZone feature is disabled. Entire OCRAM_L2 space is available for all access types (secure/non-secure/user/supervisor). 1 The TrustZone feature is enabled. Access to address in the range specified by [ENDADDR:STARTADDR] follows the execution mode access policy described in CSU chapter.
2 SEC_ERR_RESP	Security error response enable for all security gaskets (on both AHB and AXI busses) 0 OKEY response 1 SLVError (default)
1 DBG_CLK_EN	ARM Debug clock enable 0 Debug turned off. 1 Debug enabled (default).
0 DBG_EN	ARM non secure (non-invasive) debug enable

*Table continues on the next page...*

**IOMUXC\_GPR10 field descriptions (continued)**

Field	Description
0	Debug turned off.
1	Debug enabled (default).

**30.4.12 GPR11 (IOMUXC\_GPR11)**

Address: 20E\_0000h base + 2Ch offset = 20E\_002Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved														LOCK_OCRAM_ L2_EN	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														OCRAM_L2_EN	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_GPR11 field descriptions**

Field	Description
31–18 Reserved	This field is reserved. Reserved
17 LOCK_OCRAM_ L2_EN	lock ocram_l2 enable bit
16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–2 Reserved	This field is reserved. Reserved
1 OCRAM_L2_EN	set to use L2 cache as ocram
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

### 30.4.13 GPR12 (IOMUXC\_GPR12)

Address: 20E\_0000h base + 30h offset = 20E\_0030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DCP_KEY_SEL	Reserved				ARMP_IPG_CLK_EN	ARMP_AHB_CLK_EN	ARMP_ATB_CLK_EN	ARMP_APB_CLK_EN	GPU_ARQOS2				GPU_AWQOS2		
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPU_ARQOS1				GPU_AWQOS1				GPU_ARQOS0				GPU_AWQOS0			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_GPR12 field descriptions

Field	Description
31 DCP_KEY_SEL	select 128bit dcp key from 256bit's key from snvs/ocotp 0 select [127:0] from snvs/ocotp key as dcp key 1 select [255:128] from snvs/ocotp key as dcp key
30–28 Reserved	This field is reserved. Reserved
27 ARMP_IPG_CLK_EN	ARM platform IPG clock enable 0 IPG clock is not running (gated). 1 IPG clock is running (enabled).
26 ARMP_AHB_CLK_EN	ARM platform AHB clock enable 0 IPG clock is not running (gated). 1 IPG clock is running (enabled).
25 ARMP_ATB_CLK_EN	ARM platform ATB clock enable 0 IPG clock is not running (gated). 1 IPG clock is running (enabled).
24 ARMP_APB_CLK_EN	ARM platform APB clock enable 0 IPG clock is not running (gated). 1 IPG clock is running (enabled).
23–20 GPU_ARQOS2	QOS control for read channel of gpu port m_g_2

Table continues on the next page...

**IOMUXC\_GPR12 field descriptions (continued)**

Field	Description
19–16 GPU_AWQOS2	QOS control for write channel of gpu port m_g_2
15–12 GPU_ARQOS1	QOS control for read channel of gpu port m_g_1
11–8 GPU_AWQOS1	QOS control for write channel of gpu port m_g_1
7–4 GPU_ARQOS0	QOS control for read channel of gpu port m_g_0
3–0 GPU_AWQOS0	QOS control for write channel of gpu port m_g_0

**30.4.14 GPR13 (IOMUXC\_GPR13)**

SATA_PHY_6	PHUG	FRUG	fast_startup	Frequency Tolerance (ppm)
000	1	1	None	780
001	2	2	None	780
010	1	4	None	6,250
011	2	4	None	6,250
1xx	Reserved			

Address: 20E\_0000h base + 34h offset = 20E\_0034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	0	SDMA_STOP_REQ	Reserved												LCDIF_RD_CACHE_SEL	Reserved	LCDIF_RD_CACHE_VAL
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

## IOMUXC Memory Map/Register Definition

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EPDC_WR_CACHE_SEL	EPDC_RD_CACHE_SEL	EPDC_WR_CACHE_VAL	EPDC_RD_CACHE_VAL	PXP_WR_CACHE_SEL	PXP_RD_CACHE_SEL	PXP_WR_CACHE_VAL	PXP_RD_CACHE_VAL	Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

### IOMUXC\_GPR13 field descriptions

Field	Description
31 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
30 SDMA_STOP_REQ	SDMA stop request 0 Stop request off. 1 Stop request on.
29–19 Reserved	This field is reserved. Reserved
18 LCDIF_RD_CACHE_SEL	This bit selects the cacheable attribute of LCDIF AXI read transactions) 0 The read transaction cacheable attribute is driven by the LCDIF core 1 The read transaction cacheable attribute is driven by LCDIF_RD_CACHE_VAL.
17 Reserved	This field is reserved.
16 LCDIF_RD_CACHE_VAL	LCDIF block cacheable attribute value of AXI read transactions The value of LCDIF_RD_CACHE_VAL is affecting the transactions only if LCDIF_RD_CACHE_SEL is set. 0 Cacheable attribute is off for read transactions. 1 Cacheable attribute is on for read transactions.
15 EPDC_WR_CACHE_SEL	This bit selects the cacheable attribute of EPDC AXI write transactions 0 The write transactions cacheable attribute is driven by the EPDC core 1 The write transactions cacheable attribute is driven by EPDC_WR_CACHE_VAL.
14 EPDC_RD_CACHE_SEL	This bit selects the cacheable attribute of EPDC AXI read transactions) 0 The read transaction cacheable attribute is driven by the EPDC core 1 The read transaction cacheable attribute is driven by EPDC_RD_CACHE_VAL.
13 EPDC_WR_CACHE_VAL	EPDC block cacheable attribute value of AXI write transactions The value of EPDC_WR_CACHE_VAL is affecting the transactions only if EPDC_WR_CACHE_SEL is set. 0 Cacheable attribute is off for write transactions. 1 Cacheable attribute is on for write transactions.
12 EPDC_RD_CACHE_VAL	EPDC block cacheable attribute value of AXI read transactions The value of EPDC_RD_CACHE_VAL is affecting the transactions only if EPDC_RD_CACHE_SEL is set.

Table continues on the next page...

**IOMUXC\_GPR13 field descriptions (continued)**

Field	Description
	0 Cacheable attribute is off for read transactions. 1 Cacheable attribute is on for read transactions.
11 PXP_WR_CACHE_SEL	This bit selects the cacheable attribute of PXP AXI write transactions 0 The write transactions cacheable attribute is driven by the PXP core 1 The write transactions cacheable attribute is driven by PXP_WR_CACHE_VAL.
10 PXP_RD_CACHE_SEL	This bit selects the cacheable attribute of PXP AXI read transactions) 0 The read transaction cacheable attribute is driven by the PXP core 1 The read transaction cacheable attribute is driven by PXP_RD_CACHE_VAL.
9 PXP_WR_CACHE_VAL	PXP block cacheable attribute value of AXI write transactions The value of PXP_WR_CACHE_VAL is affecting the transactions only if PXP_WR_CACHE_SEL is set. 0 Cacheable attribute is off for write transactions. 1 Cacheable attribute is on for write transactions.
8 PXP_RD_CACHE_VAL	PXP block cacheable attribute value of AXI read transactions The value of PXP_RD_CACHE_VAL is affecting the transactions only if PXP_RD_CACHE_SEL is set. 0 Cacheable attribute is off for read transactions. 1 Cacheable attribute is on for read transactions.
7–0 Reserved	This field is reserved. Reserved

### 30.4.15 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_MCLK)

Address: 20E\_0000h base + 4Ch offset = 20E\_004Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

**IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_MCLK field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality.

*Table continues on the next page...*

**IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_MCLK field descriptions (continued)**

Field	Description
	1 <b>ENABLED</b> — Force input path of pad AUD_MCLK. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field. Select 1 of 7 iomux modes to be used for pad: AUD_MCLK. NOTE: Pad AUD_MCLK is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal AUDIO_CLK_OUT. 001 <b>ALT1</b> — Select signal PWM4_OUT. 010 <b>ALT2</b> — Select signal ECSPI3_RDY. - Configure register IOMUXC_ECSPI3_DATAREADY_B_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal FEC_MDC. 100 <b>ALT4</b> — Select signal WDOG2_RESET_B_DEB. 101 <b>ALT5</b> — Select signal GPIO1_IO06. 110 <b>ALT6</b> — Select signal SPDIF_EXT_CLK. - Configure register IOMUXC_SPDIF_TX_CLK2_SELECT_INPUT for mode ALT6.

### 30.4.16 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_RXC)

Address: 20E\_0000h base + 50h offset = 20E\_0050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

**IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_RXC field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field. Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad AUD_RXC. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).

*Table continues on the next page...*



**IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_RXC field descriptions (continued)**

Field	Description
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	<p>MUX Mode Select Field.</p> <p>Select 1 of 7 iomux modes to be used for pad: AUD_RXC.</p> <p>NOTE: Pad AUD_RXC is involved in Daisy Chain.</p> <p>000 <b>ALT0</b> — Select signal AUD3_RXC.</p> <p>001 <b>ALT1</b> — Select signal I2C1_SDA.</p> <p>- Configure register IOMUXC_I2C1_SDA_IN_SELECT_INPUT for mode ALT1.</p> <p>010 <b>ALT2</b> — Select signal UART3_TX_DATA.</p> <p>- Configure register IOMUXC_UART3_UART_RX_DATA_SELECT_INPUT for mode ALT2.</p> <p>011 <b>ALT3</b> — Select signal FEC_TX_CLK.</p> <p>- Configure register IOMUXC_FEC_FEC_TX_CLK_SELECT_INPUT for mode ALT3.</p> <p>100 <b>ALT4</b> — Select signal I2C3_SDA.</p> <p>- Configure register IOMUXC_I2C3_SDA_IN_SELECT_INPUT for mode ALT4.</p> <p>101 <b>ALT5</b> — Select signal GPIO1_IO01.</p> <p>110 <b>ALT6</b> — Select signal ECSPI3_SS1.</p> <p>- Configure register IOMUXC_ECSPI3_SS1_SELECT_INPUT for mode ALT6.</p>

### 30.4.17 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_RXD)

Address: 20E\_0000h base + 54h offset = 20E\_0054h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

**IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_RXD field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	<p>Software Input On Field.</p> <p>Force the selected mux mode input path no matter of MUX_MODE functionality.</p>

*Table continues on the next page...*

**IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_RXD field descriptions (continued)**

Field	Description
	1 <b>ENABLED</b> — Force input path of pad AUD_RXD. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: AUD_RXD. NOTE: Pad AUD_RXD is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal AUD3_RXD. 001 <b>ALT1</b> — Select signal ECSPI3_MOSI. - Configure register IOMUXC_ECSPI3_MOSI_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal UART4_RX_DATA. - Configure register IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal FEC_RX_ER. - Configure register IOMUXC_FEC_FEC_RX_ER_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SD1_LCTL. 101 <b>ALT5</b> — Select signal GPIO1_IO02.

### 30.4.18 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_RXFS)

Address: 20E\_0000h base + 58h offset = 20E\_0058h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

**IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_RXFS field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad AUD_RXFS. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).

*Table continues on the next page...*

**IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_RXFS field descriptions (continued)**

Field	Description
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	<p>MUX Mode Select Field.</p> <p>Select 1 of 7 iomux modes to be used for pad: AUD_RXFS.</p> <p>NOTE: Pad AUD_RXFS is involved in Daisy Chain.</p> <p>000 <b>ALT0</b> — Select signal AUD3_RXFS.</p> <p>001 <b>ALT1</b> — Select signal I2C1_SCL.</p> <p>- Configure register IOMUXC_I2C1_SCL_IN_SELECT_INPUT for mode ALT1.</p> <p>010 <b>ALT2</b> — Select signal UART3_RX_DATA.</p> <p>- Configure register IOMUXC_UART3_UART_RX_DATA_SELECT_INPUT for mode ALT2.</p> <p>011 <b>ALT3</b> — Select signal FEC_MDIO.</p> <p>- Configure register IOMUXC_FEC_FEC_MDI_SELECT_INPUT for mode ALT3.</p> <p>100 <b>ALT4</b> — Select signal I2C3_SCL.</p> <p>- Configure register IOMUXC_I2C3_SCL_IN_SELECT_INPUT for mode ALT4.</p> <p>101 <b>ALT5</b> — Select signal GPIO1_IO00.</p> <p>110 <b>ALT6</b> — Select signal ECSPI3_SS0.</p> <p>- Configure register IOMUXC_ECSPI3_SS0_SELECT_INPUT for mode ALT6.</p>

### 30.4.19 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_TXC)

Address: 20E\_0000h base + 5Ch offset = 20E\_005Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

**IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_TXC field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	<p>Software Input On Field.</p> <p>Force the selected mux mode input path no matter of MUX_MODE functionality.</p>

*Table continues on the next page...*

**IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_TXC field descriptions (continued)**

Field	Description
	1 <b>ENABLED</b> — Force input path of pad AUD_TXC. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: AUD_TXC. NOTE: Pad AUD_TXC is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal AUD3_TXC. 001 <b>ALT1</b> — Select signal ECSPI3_MISO. - Configure register IOMUXC_ECSPI3_MISO_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal UART4_TX_DATA. - Configure register IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal FEC_RX_DV. - Configure register IOMUXC_FEC_FEC_RX_DV_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SD2_LCTL. 101 <b>ALT5</b> — Select signal GPIO1_IO03.

### 30.4.20 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_TXD)

Address: 20E\_0000h base + 60h offset = 20E\_0060h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

**IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_TXD field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad AUD_TXD. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).

*Table continues on the next page...*

**IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_TXD field descriptions (continued)**

Field	Description
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	<p>MUX Mode Select Field.</p> <p>Select 1 of 6 iomux modes to be used for pad: AUD_TXD.</p> <p>NOTE: Pad AUD_TXD is involved in Daisy Chain.</p> <p>000 <b>ALT0</b> — Select signal AUD3_TXD.</p> <p>001 <b>ALT1</b> — Select signal ECSPI3_SCLK.</p> <p>- Configure register IOMUXC_ECSPi3_CSPI_CLK_IN_SELECT_INPUT for mode ALT1.</p> <p>010 <b>ALT2</b> — Select signal UART4_CTS_B.</p> <p>- Configure register IOMUXC_UART4_UART_RTS_B_SELECT_INPUT for mode ALT2.</p> <p>011 <b>ALT3</b> — Select signal FEC_TX_DATA0.</p> <p>100 <b>ALT4</b> — Select signal SD4_LCTL.</p> <p>101 <b>ALT5</b> — Select signal GPIO1_IO05.</p>

### 30.4.21 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_TXFS)

Address: 20E\_0000h base + 64h offset = 20E\_0064h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

**IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_TXFS field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	<p>Software Input On Field.</p> <p>Force the selected mux mode input path no matter of MUX_MODE functionality.</p> <p>1 <b>ENABLED</b> — Force input path of pad AUD_TXFS.</p> <p>0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).</p>
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.

Table continues on the next page...

**IOMUXC\_SW\_MUX\_CTL\_PAD\_AUD\_TXFS field descriptions (continued)**

Field	Description
	Select 1 of 6 iomux modes to be used for pad: AUD_TXFS. NOTE: Pad AUD_TXFS is involved in Daisy Chain.
000	<b>ALT0</b> — Select signal AUD3_TXFS.
001	<b>ALT1</b> — Select signal PWM3_OUT.
010	<b>ALT2</b> — Select signal UART4_RTS_B. - Configure register IOMUXC_UART4_UART_RTS_B_SELECT_INPUT for mode ALT2.
011	<b>ALT3</b> — Select signal FEC_RX_DATA1. - Configure register IOMUXC_FEC_FEC_RX_DATA1_SELECT_INPUT for mode ALT3.
100	<b>ALT4</b> — Select signal SD3_LCTL.
101	<b>ALT5</b> — Select signal GPIO1_IO04.

### 30.4.22 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSP11\_MISO)

Address: 20E\_0000h base + 68h offset = 20E\_0068h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

**IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSP11\_MISO field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad ECSP11_MISO. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: ECSP11_MISO. NOTE: Pad ECSP11_MISO is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal ECSP11_MISO.

*Table continues on the next page...*

**IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSP11\_MISO field descriptions (continued)**

Field	Description
001	- Configure register IOMUXC_ECSP11_MISO_SELECT_INPUT for mode ALT0. <b>ALT1</b> — Select signal AUD4_TXFS.
010	- Configure register IOMUXC_AUD4_INPUT_TXFS_AMX_SELECT_INPUT for mode ALT1. <b>ALT2</b> — Select signal UART5_RTS_B.
011	- Configure register IOMUXC_UART5_UART_RTS_B_SELECT_INPUT for mode ALT2. <b>ALT3</b> — Select signal EPDC_BDR0.
100	<b>ALT4</b> — Select signal SD2_WP.
	- Configure register IOMUXC_USDHC2_WP_ON_SELECT_INPUT for mode ALT4.
101	<b>ALT5</b> — Select signal GPIO4_IO10.

### 30.4.23 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSP11\_MOSI)

Address: 20E\_0000h base + 6Ch offset = 20E\_006Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										SION		0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

**IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSP11\_MOSI field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad ECSP11_MOSI. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: ECSP11_MOSI. NOTE: Pad ECSP11_MOSI is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal ECSP11_MOSI.  - Configure register IOMUXC_ECSP11_MOSI_SELECT_INPUT for mode ALT0.

*Table continues on the next page...*

**IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSP11\_MOSI field descriptions (continued)**

Field	Description
001 <b>ALT1</b>	— Select signal AUD4_TXC. - Configure register IOMUXC_AUD4_INPUT_TXCLK_AMX_SELECT_INPUT for mode ALT1.
010 <b>ALT2</b>	— Select signal UART5_TX_DATA. - Configure register IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT for mode ALT2.
011 <b>ALT3</b>	— Select signal EPDC_VCOM1.
100 <b>ALT4</b>	— Select signal SD2_VSELECT.
101 <b>ALT5</b>	— Select signal GPIO4_IO09.

### 30.4.24 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSP11\_SCLK)

Address: 20E\_0000h base + 70h offset = 20E\_0070h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

**IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSP11\_SCLK field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad ECSP11_SCLK. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: ECSP11_SCLK.  NOTE: Pad ECSP11_SCLK is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal ECSP11_SCLK. - Configure register IOMUXC_ECSP11_CSPI_CLK_IN_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal AUD4_TXD. - Configure register IOMUXC_AUD4_INPUT_DB_AMX_SELECT_INPUT for mode ALT1.

*Table continues on the next page...*



**IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSP11\_SCLK field descriptions (continued)**

Field	Description
010	<b>ALT2</b> — Select signal UART5_RX_DATA. - Configure register IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT for mode ALT2.
011	<b>ALT3</b> — Select signal EPDC_VCOM0.
100	<b>ALT4</b> — Select signal SD2_RESET.
101	<b>ALT5</b> — Select signal GPIO4_IO08.
110	<b>ALT6</b> — Select signal USB_OTG2_OC. - Configure register IOMUXC_USB_OTG2_OC_SELECT_INPUT for mode ALT6.

### 30.4.25 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSP11\_SS0)

Address: 20E\_0000h base + 74h offset = 20E\_0074h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

**IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSP11\_SS0 field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad ECSP11_SS0. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: ECSP11_SS0.  NOTE: Pad ECSP11_SS0 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal ECSP11_SS0. - Configure register IOMUXC_ECSP11_SS0_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal AUD4_RXD. - Configure register IOMUXC_AUD4_INPUT_DA_AMX_SELECT_INPUT for mode ALT1.

Table continues on the next page...

**IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSPi1\_SS0 field descriptions (continued)**

Field	Description
010 <b>ALT2</b>	— Select signal UART5_CTS_B. - Configure register IOMUXC_UART5_UART_RTS_B_SELECT_INPUT for mode ALT2.
011 <b>ALT3</b>	— Select signal EPDC_BDR1.
100 <b>ALT4</b>	— Select signal SD2_CD_B. - Configure register IOMUXC_USDHC2_CARD_DET_SELECT_INPUT for mode ALT4.
101 <b>ALT5</b>	— Select signal GPIO4_IO11.
110 <b>ALT6</b>	— Select signal USB_OTG2_PWR.

### 30.4.26 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSPi2\_MISO)

Address: 20E\_0000h base + 78h offset = 20E\_0078h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

**IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSPi2\_MISO field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad ECSPi2_MISO. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: ECSPi2_MISO.  NOTE: Pad ECSPi2_MISO is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal ECSPi2_MISO. - Configure register IOMUXC_ECSPi2_MISO_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal SDMA_EXT_EVENT0. 010 <b>ALT2</b> — Select signal UART3_RTS_B.

*Table continues on the next page...*

**IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSPi2\_MISO field descriptions (continued)**

Field	Description
011	- Configure register IOMUXC_UART3_UART_RTS_B_SELECT_INPUT for mode ALT2. <b>ALT3</b> — Select signal CSI_MCLK.
100	<b>ALT4</b> — Select signal SD1_WP. - Configure register IOMUXC_USDHC1_WP_ON_SELECT_INPUT for mode ALT4.
101	<b>ALT5</b> — Select signal GPIO4_IO14.
110	<b>ALT6</b> — Select signal USB_OTG1_OC. - Configure register IOMUXC_USB_OTG1_OC_SELECT_INPUT for mode ALT6.

### 30.4.27 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSPi2\_MOSI)

Address: 20E\_0000h base + 7Ch offset = 20E\_007Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

**IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSPi2\_MOSI field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad ECSPi2_MOSI. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: ECSPi2_MOSI.  NOTE: Pad ECSPi2_MOSI is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal ECSPi2_MOSI. - Configure register IOMUXC_ECSPi2_MOSI_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal SDMA_EXT_EVENT1. 010 <b>ALT2</b> — Select signal UART3_TX_DATA.

*Table continues on the next page...*

**IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSPi2\_MOSI field descriptions (continued)**

Field	Description
011	<ul style="list-style-type: none"> <li>- Configure register IOMUXC_UART3_UART_RX_DATA_SELECT_INPUT for mode ALT2.</li> <li><b>ALT3</b> — Select signal CSI_HSYNC.</li> <li>- Configure register IOMUXC_CSI_CSI_HSYNC_SELECT_INPUT for mode ALT3.</li> </ul>
100	<b>ALT4</b> — Select signal SD1_VSELECT.
101	<b>ALT5</b> — Select signal GPIO4_IO13.

### 30.4.28 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSPi2\_SCLK)

Address: 20E\_0000h base + 80h offset = 20E\_0080h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0										SION		0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

**IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSPi2\_SCLK field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	<p>Software Input On Field.</p> <p>Force the selected mux mode input path no matter of MUX_MODE functionality.</p> <p>1 <b>ENABLED</b> — Force input path of pad ECSPi2_SCLK.</p> <p>0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).</p>
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	<p>MUX Mode Select Field.</p> <p>Select 1 of 7 iomux modes to be used for pad: ECSPi2_SCLK.</p> <p>NOTE: Pad ECSPi2_SCLK is involved in Daisy Chain.</p> <p>000 <b>ALT0</b> — Select signal ECSPi2_SCLK.</p> <ul style="list-style-type: none"> <li>- Configure register IOMUXC_ECSPi2_CSPI_CLK_IN_SELECT_INPUT for mode ALT0.</li> </ul> <p>001 <b>ALT1</b> — Select signal SPDIF_EXT_CLK.</p> <ul style="list-style-type: none"> <li>- Configure register IOMUXC_SPDIF_TX_CLK2_SELECT_INPUT for mode ALT1.</li> </ul> <p>010 <b>ALT2</b> — Select signal UART3_RX_DATA.</p> <ul style="list-style-type: none"> <li>- Configure register IOMUXC_UART3_UART_RX_DATA_SELECT_INPUT for mode ALT2.</li> </ul>

*Table continues on the next page...*

**IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSPi2\_SCLK field descriptions (continued)**

Field	Description
011	<b>ALT3</b> — Select signal CSI_PIXCLK. - Configure register IOMUXC_CSI_CSI_PIXCLK_SELECT_INPUT for mode ALT3.
100	<b>ALT4</b> — Select signal SD1_RESET.
101	<b>ALT5</b> — Select signal GPIO4_IO12.
110	<b>ALT6</b> — Select signal USB_OTG2_OC. - Configure register IOMUXC_USB_OTG2_OC_SELECT_INPUT for mode ALT6.

### 30.4.29 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSPi2\_SS0)

Address: 20E\_0000h base + 84h offset = 20E\_0084h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0											SION	0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

**IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSPi2\_SS0 field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad ECSPi2_SS0. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: ECSPi2_SS0.  NOTE: Pad ECSPi2_SS0 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal ECSPi2_SS0. - Configure register IOMUXC_ECSPi2_SS0_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal ECSPi1_SS3. - Configure register IOMUXC_ECSPi1_SS3_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal UART3_CTS_B.

Table continues on the next page...

**IOMUXC\_SW\_MUX\_CTL\_PAD\_ECSPi2\_SS0 field descriptions (continued)**

Field	Description
011	- Configure register IOMUXC_UART3_UART_RTS_B_SELECT_INPUT for mode ALT2. <b>ALT3</b> — Select signal CSI_VSYNC.
100	- Configure register IOMUXC_CSI_CSI_VSYNC_SELECT_INPUT for mode ALT3. <b>ALT4</b> — Select signal SD1_CD_B.
101	- Configure register IOMUXC_USDHC1_CARD_DET_SELECT_INPUT for mode ALT4. <b>ALT5</b> — Select signal GPIO4_IO15.
110	<b>ALT6</b> — Select signal USB_OTG1_PWR.

### 30.4.30 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_BDR0)

Address: 20E\_0000h base + 88h offset = 20E\_0088h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

**IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_BDR0 field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_BDR0. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_BDR0.  NOTE: Pad EPDC_BDR0 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_BDR0. 001 <b>ALT1</b> — Select signal SD4_CLK. - Configure register IOMUXC_USDHC4_CARD_CLK_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal UART3_RTS_B.

*Table continues on the next page...*

**IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_BDR0 field descriptions (continued)**

Field	Description
	- Configure register IOMUXC_UART3_UART_RTS_B_SELECT_INPUT for mode ALT2.
011	<b>ALT3</b> — Select signal EIM_ADDR26.
100	<b>ALT4</b> — Select signal SPDC_RL.
101	<b>ALT5</b> — Select signal GPIO2_IO05.
110	<b>ALT6</b> — Select signal EPDC_SDCE7.

### 30.4.31 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_BDR1)

Address: 20E\_0000h base + 8Ch offset = 20E\_008Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

**IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_BDR1 field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_BDR1. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_BDR1.  NOTE: Pad EPDC_BDR1 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_BDR1. 001 <b>ALT1</b> — Select signal SD4_CMD. - Configure register IOMUXC_USDHC4_CMD_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal UART3_CTS_B. - Configure register IOMUXC_UART3_UART_RTS_B_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_CRE. 100 <b>ALT4</b> — Select signal SPDC_UD.

*Table continues on the next page...*

**IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_BDR1 field descriptions (continued)**

Field	Description
101 <b>ALT5</b>	Select signal GPIO2_IO06.
110 <b>ALT6</b>	Select signal EPDC_SDCE8.

### 30.4.32 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA00)

Address: 20E\_0000h base + 90h offset = 20E\_0090h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

**IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA00 field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_D0. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: EPDC_D0.  NOTE: Pad EPDC_D0 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_DATA00. 001 <b>ALT1</b> — Select signal ECSPI4_MOSI. - Configure register IOMUXC_ECSPI4_MOSI_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA24. 011 <b>ALT3</b> — Select signal CSI_DATA00. - Configure register IOMUXC_CSI_CSI_DATA00_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SPDC_DATA00. 101 <b>ALT5</b> — Select signal GPIO1_IO07.



### 30.4.33 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA01)

Address: 20E\_0000h base + 94h offset = 20E\_0094h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA01 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_D1. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: EPDC_D1.  NOTE: Pad EPDC_D1 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_DATA01. 001 <b>ALT1</b> — Select signal ECSPI4_MISO. - Configure register IOMUXC_ECSPI4_MISO_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA25. 011 <b>ALT3</b> — Select signal CSI_DATA01. - Configure register IOMUXC_CSI_CSI_DATA01_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SPDC_DATA01. 101 <b>ALT5</b> — Select signal GPIO1_IO08.

### 30.4.34 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA10)

Address: 20E\_0000h base + 98h offset = 20E\_0098h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA10 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_D10. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_D10.  NOTE: Pad EPDC_D10 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_DATA10. 001 <b>ALT1</b> — Select signal ECSPI3_SS0. - Configure register IOMUXC_ECSPI3_SS0_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal EPDC_PWR_CTRL2. 011 <b>ALT3</b> — Select signal EIM_ADDR18. 100 <b>ALT4</b> — Select signal SPDC_DATA10. 101 <b>ALT5</b> — Select signal GPIO1_IO17. 110 <b>ALT6</b> — Select signal SD4_WP. - Configure register IOMUXC_USDHC4_WP_ON_SELECT_INPUT for mode ALT6.

### 30.4.35 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA11)

Address: 20E\_0000h base + 9Ch offset = 20E\_009Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA11 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_D11. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_D11.  NOTE: Pad EPDC_D11 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_DATA11. 001 <b>ALT1</b> — Select signal ECSPI3_SCLK. - Configure register IOMUXC_ECSPI3_CSPI_CLK_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal EPDC_PWR_CTRL3. 011 <b>ALT3</b> — Select signal EIM_ADDR19. 100 <b>ALT4</b> — Select signal SPDC_DATA11. 101 <b>ALT5</b> — Select signal GPIO1_IO18. 110 <b>ALT6</b> — Select signal SD4_CD_B. - Configure register IOMUXC_USDHC4_CARD_DET_SELECT_INPUT for mode ALT6.

### 30.4.36 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA12)

Address: 20E\_0000h base + A0h offset = 20E\_00A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA12 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_D12. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_D12.  NOTE: Pad EPDC_D12 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_DATA12. 001 <b>ALT1</b> — Select signal UART2_RX_DATA. - Configure register IOMUXC_UART2_UART_RX_DATA_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal EPDC_PWR_COM. 011 <b>ALT3</b> — Select signal EIM_ADDR20. 100 <b>ALT4</b> — Select signal SPDC_DATA12. 101 <b>ALT5</b> — Select signal GPIO1_IO19. 110 <b>ALT6</b> — Select signal ECSPI3_SS1. - Configure register IOMUXC_ECSPI3_SS1_SELECT_INPUT for mode ALT6.

### 30.4.37 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA13)

Address: 20E\_0000h base + A4h offset = 20E\_00A4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA13 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_D13. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_D13.  NOTE: Pad EPDC_D13 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_DATA13. 001 <b>ALT1</b> — Select signal UART2_TX_DATA. - Configure register IOMUXC_UART2_UART_RX_DATA_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal EPDC_PWR_IRQ. - Configure register IOMUXC_EPDC_EPDC_PWR_IRQ_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_ADDR21. 100 <b>ALT4</b> — Select signal SPDC_DATA13. 101 <b>ALT5</b> — Select signal GPIO1_IO20. 110 <b>ALT6</b> — Select signal ECSPI3_SS2. - Configure register IOMUXC_ECSPI3_SS2_SELECT_INPUT for mode ALT6.

### 30.4.38 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA14)

Address: 20E\_0000h base + A8h offset = 20E\_00A8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA14 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_D14. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_D14.  NOTE: Pad EPDC_D14 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_DATA14. 001 <b>ALT1</b> — Select signal UART2_RTS_B. - Configure register IOMUXC_UART2_UART_RTS_B_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal EPDC_PWR_STAT. - Configure register IOMUXC_EPDC_EPDC_PWR_STAT_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_ADDR22. 100 <b>ALT4</b> — Select signal SPDC_DATA14. 101 <b>ALT5</b> — Select signal GPIO1_IO21. 110 <b>ALT6</b> — Select signal ECSPI3_SS3. - Configure register IOMUXC_ECSPi3_SS3_SELECT_INPUT for mode ALT6.

### 30.4.39 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA15)

Address: 20E\_0000h base + ACh offset = 20E\_00ACh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA15 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_D15. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_D15.  NOTE: Pad EPDC_D15 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_DATA15. 001 <b>ALT1</b> — Select signal UART2_CTS_B. - Configure register IOMUXC_UART2_UART_RTS_B_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal EPDC_PWR_WAKE. 011 <b>ALT3</b> — Select signal EIM_ADDR23. 100 <b>ALT4</b> — Select signal SPDC_DATA15. 101 <b>ALT5</b> — Select signal GPIO1_IO22. 110 <b>ALT6</b> — Select signal ECSPI3_RDY. - Configure register IOMUXC_ECSPI3_DATAREADY_B_SELECT_INPUT for mode ALT6.

### 30.4.40 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA02)

Address: 20E\_0000h base + B0h offset = 20E\_00B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA02 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_D2. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: EPDC_D2.  NOTE: Pad EPDC_D2 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_DATA02. 001 <b>ALT1</b> — Select signal ECSPi4_SS0. - Configure register IOMUXC_ECSPi4_SS0_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA26. 011 <b>ALT3</b> — Select signal CSI_DATA02. - Configure register IOMUXC_CSI_CSI_DATA02_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SPDC_DATA02. 101 <b>ALT5</b> — Select signal GPIO1_IO09.



### 30.4.41 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA03)

Address: 20E\_0000h base + B4h offset = 20E\_00B4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA03 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_D3. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: EPDC_D3.  NOTE: Pad EPDC_D3 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_DATA03. 001 <b>ALT1</b> — Select signal ECSPI4_SCLK. - Configure register IOMUXC_ECSPI4_CSPI_CLK_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA27. 011 <b>ALT3</b> — Select signal CSI_DATA03. - Configure register IOMUXC_CSI_CSI_DATA03_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SPDC_DATA03. 101 <b>ALT5</b> — Select signal GPIO1_IO10.

### 30.4.42 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA04)

Address: 20E\_0000h base + B8h offset = 20E\_00B8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA04 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_D4. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: EPDC_D4.  NOTE: Pad EPDC_D4 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_DATA04. 001 <b>ALT1</b> — Select signal ECSPI4_SS1. - Configure register IOMUXC_ECSPI4_SS1_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA28. 011 <b>ALT3</b> — Select signal CSI_DATA04. - Configure register IOMUXC_CSI_CSI_DATA04_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SPDC_DATA04. 101 <b>ALT5</b> — Select signal GPIO1_IO11.

### 30.4.43 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA05)

Address: 20E\_0000h base + BCh offset = 20E\_00BCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA05 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_D5. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: EPDC_D5.  NOTE: Pad EPDC_D5 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_DATA05. 001 <b>ALT1</b> — Select signal ECSPi4_SS2. - Configure register IOMUXC_ECSPi4_SS2_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA29. 011 <b>ALT3</b> — Select signal CSI_DATA05. - Configure register IOMUXC_CSI_CSI_DATA05_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SPDC_DATA05. 101 <b>ALT5</b> — Select signal GPIO1_IO12.

### 30.4.44 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA06)

Address: 20E\_0000h base + C0h offset = 20E\_00C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA06 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_D6. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: EPDC_D6.  NOTE: Pad EPDC_D6 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_DATA06. 001 <b>ALT1</b> — Select signal ECSPI4_SS3. 010 <b>ALT2</b> — Select signal LCD_DATA30. 011 <b>ALT3</b> — Select signal CSI_DATA06.  - Configure register IOMUXC_CSI_CSI_DATA06_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SPDC_DATA06. 101 <b>ALT5</b> — Select signal GPIO1_IO13.

### 30.4.45 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA07)

Address: 20E\_0000h base + C4h offset = 20E\_00C4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA07 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_D7. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: EPDC_D7.  NOTE: Pad EPDC_D7 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_DATA07. 001 <b>ALT1</b> — Select signal ECSPI4_RDY. 010 <b>ALT2</b> — Select signal LCD_DATA31. 011 <b>ALT3</b> — Select signal CSI_DATA07.  - Configure register IOMUXC_CSI_CSI_DATA07_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SPDC_DATA07. 101 <b>ALT5</b> — Select signal GPIO1_IO14.

### 30.4.46 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA08)

Address: 20E\_0000h base + C8h offset = 20E\_00C8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA08 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_D8. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_D8.  NOTE: Pad EPDC_D8 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_DATA08. 001 <b>ALT1</b> — Select signal ECSPI3_MOSI. - Configure register IOMUXC_ECSPI3_MOSI_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal EPDC_PWR_CTRL0. 011 <b>ALT3</b> — Select signal EIM_ADDR16. 100 <b>ALT4</b> — Select signal SPDC_DATA08. 101 <b>ALT5</b> — Select signal GPIO1_IO15. 110 <b>ALT6</b> — Select signal SD4_RESET.

### 30.4.47 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA09)

Address: 20E\_0000h base + CCh offset = 20E\_00CCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_DATA09 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_D9. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_D9.  NOTE: Pad EPDC_D9 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_DATA09. 001 <b>ALT1</b> — Select signal ECSPI3_MISO. - Configure register IOMUXC_ECSPI3_MISO_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal EPDC_PWR_CTRL1. 011 <b>ALT3</b> — Select signal EIM_ADDR17. 100 <b>ALT4</b> — Select signal SPDC_DATA09. 101 <b>ALT5</b> — Select signal GPIO1_IO16. 110 <b>ALT6</b> — Select signal SD4_VSELECT.

### 30.4.48 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_GDCLK)

Address: 20E\_0000h base + D0h offset = 20E\_00D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_GDCLK field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_GDCLK. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_GDCLK.  NOTE: Pad EPDC_GDCLK is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_GDCLK. 001 <b>ALT1</b> — Select signal ECSPI2_SS2. 010 <b>ALT2</b> — Select signal SPDC_YCKR. 011 <b>ALT3</b> — Select signal CSI_PIXCLK.  - Configure register IOMUXC_CSI_CSI_PIXCLK_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SPDC_YCKL. 101 <b>ALT5</b> — Select signal GPIO1_IO31. 110 <b>ALT6</b> — Select signal SD2_RESET.



### 30.4.49 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_GDOE)

Address: 20E\_0000h base + D4h offset = 20E\_00D4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_GDOE field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_GDOE. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_GDOE.  NOTE: Pad EPDC_GDOE is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_GDOE. 001 <b>ALT1</b> — Select signal ECSPI2_SS3. 010 <b>ALT2</b> — Select signal SPDC_YOER. 011 <b>ALT3</b> — Select signal CSI_HSYNC. - Configure register IOMUXC_CSI_CSI_HSYNC_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SPDC_YOEL. 101 <b>ALT5</b> — Select signal GPIO2_IO00. 110 <b>ALT6</b> — Select signal SD2_VSELECT.

### 30.4.50 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_GDRL)

Address: 20E\_0000h base + D8h offset = 20E\_00D8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_GDRL field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_GDRL. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_GDRL.  NOTE: Pad EPDC_GDRL is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_GDRL. 001 <b>ALT1</b> — Select signal ECSPI2_RDY. 010 <b>ALT2</b> — Select signal SPDC_YDIOUR. 011 <b>ALT3</b> — Select signal CSI_MCLK. 100 <b>ALT4</b> — Select signal SPDC_YDIOUL. 101 <b>ALT5</b> — Select signal GPIO2_IO01. 110 <b>ALT6</b> — Select signal SD2_WP.  - Configure register IOMUXC_USDHC2_WP_ON_SELECT_INPUT for mode ALT6.

### 30.4.51 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_GDSP)

Address: 20E\_0000h base + DCh offset = 20E\_00DCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_GDSP field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_GDSP. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_GDSP.  NOTE: Pad EPDC_GDSP is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_GDSP. 001 <b>ALT1</b> — Select signal PWM4_OUT. 010 <b>ALT2</b> — Select signal SPDC_YDIODR. 011 <b>ALT3</b> — Select signal CSI_VSYNC. - Configure register IOMUXC_CSI_CSI_VSYNC_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SPDC_YDIODL. 101 <b>ALT5</b> — Select signal GPIO2_IO02. 110 <b>ALT6</b> — Select signal SD2_CD_B. - Configure register IOMUXC_USDHC2_CARD_DET_SELECT_INPUT for mode ALT6.

### 30.4.52 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_PWR\_COM)

Address: 20E\_0000h base + E0h offset = 20E\_00E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_PWR\_COM field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_PWR_COM. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_PWR_COM.  NOTE: Pad EPDC_PWR_COM is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_PWR_COM. 001 <b>ALT1</b> — Select signal SD4_DATA0. - Configure register IOMUXC_USDHC4_DATA0_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA20. - Configure register IOMUXC_LCD_DATA20_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_BCLK. 100 <b>ALT4</b> — Select signal USB_OTG1_ID. - Configure register IOMUXC_ANALOG_USB_OTG_ID_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO2_IO11. 110 <b>ALT6</b> — Select signal SD3_RESET.

### 30.4.53 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_PWR\_CTRL0)

Address: 20E\_0000h base + E4h offset = 20E\_00E4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_PWR\_CTRL0 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_PWRCTRL0. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_PWRCTRL0.  NOTE: Pad EPDC_PWRCTRL0 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_PWR_CTRL0. 001 <b>ALT1</b> — Select signal AUD5_RXC. - Configure register IOMUXC_AUD5_INPUT_RXCLK_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA16. - Configure register IOMUXC_LCD_DATA16_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_RW. 100 <b>ALT4</b> — Select signal SPDC_YCKL. 101 <b>ALT5</b> — Select signal GPIO2_IO07. 110 <b>ALT6</b> — Select signal SD4_RESET.

### 30.4.54 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_PWR\_CTRL1)

Address: 20E\_0000h base + E8h offset = 20E\_00E8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_PWR\_CTRL1 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_PWRCTRL1. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_PWRCTRL1.  NOTE: Pad EPDC_PWRCTRL1 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_PWR_CTRL1. 001 <b>ALT1</b> — Select signal AUD5_TXFS. - Configure register IOMUXC_AUD5_INPUT_TXFS_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA17. - Configure register IOMUXC_LCD_DATA17_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_OE_B. 100 <b>ALT4</b> — Select signal SPDC_YOEL. 101 <b>ALT5</b> — Select signal GPIO2_IO08. 110 <b>ALT6</b> — Select signal SD4_VSELECT.

### 30.4.55 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_PWR\_CTRL2)

Address: 20E\_0000h base + ECh offset = 20E\_00ECh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_PWR\_CTRL2 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_PWRCTRL2. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_PWRCTRL2.  NOTE: Pad EPDC_PWRCTRL2 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_PWR_CTRL2. 001 <b>ALT1</b> — Select signal AUD5_TXD. - Configure register IOMUXC_AUD5_INPUT_DB_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA18. - Configure register IOMUXC_LCD_DATA18_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_CS0_B. 100 <b>ALT4</b> — Select signal SPDC_YDIOUL. 101 <b>ALT5</b> — Select signal GPIO2_IO09. 110 <b>ALT6</b> — Select signal SD4_WP. - Configure register IOMUXC_USDHC4_WP_ON_SELECT_INPUT for mode ALT6.

### 30.4.56 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_PWR\_CTRL3)

Address: 20E\_0000h base + F0h offset = 20E\_00F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_PWR\_CTRL3 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_PWRCTRL3. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_PWRCTRL3.  NOTE: Pad EPDC_PWRCTRL3 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_PWR_CTRL3. 001 <b>ALT1</b> — Select signal AUD5_TXC. - Configure register IOMUXC_AUD5_INPUT_TXCLK_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA19. - Configure register IOMUXC_LCD_DATA19_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_CS1_B. 100 <b>ALT4</b> — Select signal SPDC_YDIODL. 101 <b>ALT5</b> — Select signal GPIO2_IO10. 110 <b>ALT6</b> — Select signal SD4_CD_B. - Configure register IOMUXC_USDHC4_CARD_DET_SELECT_INPUT for mode ALT6.



### 30.4.57 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_PWR\_IRQ)

Address: 20E\_0000h base + F4h offset = 20E\_00F4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_PWR\_IRQ field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_PWRINT. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_PWRINT.  NOTE: Pad EPDC_PWRINT is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_PWR_IRQ. - Configure register IOMUXC_EPDC_EPDC_PWR_IRQ_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal SD4_DATA1. - Configure register IOMUXC_USDHC4_DATA1_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA21. - Configure register IOMUXC_LCD_DATA21_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_ACLK_FREERUN. 100 <b>ALT4</b> — Select signal USB_OTG2_ID. - Configure register IOMUXC_ANALOG_USB_H1_ID_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO2_IO12. 110 <b>ALT6</b> — Select signal SD3_VSELECT.

### 30.4.58 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_PWR\_STAT)

Address: 20E\_0000h base + F8h offset = 20E\_00F8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_PWR\_STAT field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_PWRSTAT. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_PWRSTAT.  NOTE: Pad EPDC_PWRSTAT is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_PWR_STAT. - Configure register IOMUXC_EPDC_EPDC_PWR_STAT_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal SD4_DATA2. - Configure register IOMUXC_USDHC4_DATA2_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA22. - Configure register IOMUXC_LCD_DATA22_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_WAIT_B. - Configure register IOMUXC_EIM_WAIT_B_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal ARM_EVENT1. 101 <b>ALT5</b> — Select signal GPIO2_IO13. 110 <b>ALT6</b> — Select signal SD3_WP. - Configure register IOMUXC_USDHC3_WP_ON_SELECT_INPUT for mode ALT6.

### 30.4.59 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_PWR\_WAKE)

Address: 20E\_0000h base + FCh offset = 20E\_00FCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_PWR\_WAKE field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_PWRWAKEUP. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_PWRWAKEUP.  NOTE: Pad EPDC_PWRWAKEUP is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_PWR_WAKE. 001 <b>ALT1</b> — Select signal SD4_DATA3. - Configure register IOMUXC_USDHC4_DATA3_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA23. - Configure register IOMUXC_LCD_DATA23_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_DTACK_B. - Configure register IOMUXC_EIM_DTACK_B_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal ARM_EVENTO. 101 <b>ALT5</b> — Select signal GPIO2_IO14. 110 <b>ALT6</b> — Select signal SD3_CD_B. - Configure register IOMUXC_USDHC3_CARD_DET_SELECT_INPUT for mode ALT6.

### 30.4.60 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_SDCE0)

Address: 20E\_0000h base + 100h offset = 20E\_0100h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_SDCE0 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_SDCE0. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: EPDC_SDCE0.  NOTE: Pad EPDC_SDCE0 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_SDCE0. 001 <b>ALT1</b> — Select signal ECSPi2_SS1. - Configure register IOMUXC_ECSPi2_SS1_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal PWM3_OUT. 011 <b>ALT3</b> — Select signal EIM_CS2_B. 100 <b>ALT4</b> — Select signal SPDC_YCKR. 101 <b>ALT5</b> — Select signal GPIO1_IO27.

### 30.4.61 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_SDCE1)

Address: 20E\_0000h base + 104h offset = 20E\_0104h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_SDCE1 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_SDCE1. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: EPDC_SDCE1.  000 <b>ALT0</b> — Select signal EPDC_SDCE1. 001 <b>ALT1</b> — Select signal WDOG2_B. 010 <b>ALT2</b> — Select signal PWM4_OUT. 011 <b>ALT3</b> — Select signal EIM_LBA_B. 100 <b>ALT4</b> — Select signal SPDC_YOER. 101 <b>ALT5</b> — Select signal GPIO1_IO28.

## 30.4.62 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_SDCE2)

Address: 20E\_0000h base + 108h offset = 20E\_0108h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_SDCE2 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_SDCE2. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: EPDC_SDCE2.  NOTE: Pad EPDC_SDCE2 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_SDCE2. 001 <b>ALT1</b> — Select signal I2C3_SCL. - Configure register IOMUXC_I2C3_SCL_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal PWM1_OUT. 011 <b>ALT3</b> — Select signal EIM_EB0_B. 100 <b>ALT4</b> — Select signal SPDC_YDIOUR. 101 <b>ALT5</b> — Select signal GPIO1_IO29.

### 30.4.63 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_SDCE3)

Address: 20E\_0000h base + 10Ch offset = 20E\_010Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_SDCE3 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_SDCE3. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: EPDC_SDCE3.  NOTE: Pad EPDC_SDCE3 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_SDCE3. 001 <b>ALT1</b> — Select signal I2C3_SDA. - Configure register IOMUXC_I2C3_SDA_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal PWM2_OUT. 011 <b>ALT3</b> — Select signal EIM_EB1_B. 100 <b>ALT4</b> — Select signal SPDC_YDIODR. 101 <b>ALT5</b> — Select signal GPIO1_IO30.

### 30.4.64 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_SDCLK)

Address: 20E\_0000h base + 110h offset = 20E\_0110h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_SDCLK field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_SDCLK. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: EPDC_SDCLK.  NOTE: Pad EPDC_SDCLK is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_SDCLK_P. 001 <b>ALT1</b> — Select signal ECSPI2_MOSI. - Configure register IOMUXC_ECSPI2_MOSI_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal I2C2_SCL. - Configure register IOMUXC_I2C2_SCL_IN_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA08. - Configure register IOMUXC_CSI_CSI_DATA08_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SPDC_CL. 101 <b>ALT5</b> — Select signal GPIO1_IO23.



### 30.4.65 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_SDLE)

Address: 20E\_0000h base + 114h offset = 20E\_0114h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_SDLE field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_SDLE. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: EPDC_SDLE.  NOTE: Pad EPDC_SDLE is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_SDLE. 001 <b>ALT1</b> — Select signal ECSPI2_MISO. - Configure register IOMUXC_ECSPI2_MISO_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal I2C2_SDA. - Configure register IOMUXC_I2C2_SDA_IN_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA09. - Configure register IOMUXC_CSI_CSI_DATA09_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SPDC_LD. 101 <b>ALT5</b> — Select signal GPIO1_IO24.

### 30.4.66 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_SDOE)

Address: 20E\_0000h base + 118h offset = 20E\_0118h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_SDOE field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_SDOE. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: EPDC_SDOE.  NOTE: Pad EPDC_SDOE is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_SDOE. 001 <b>ALT1</b> — Select signal ECSPi2_SS0. - Configure register IOMUXC_ECSPi2_SS0_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal SPDC_XDIOR. 011 <b>ALT3</b> — Select signal CSI_DATA10. - Configure register IOMUXC_CSI_CSI_DATA10_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SPDC_XDIOL. 101 <b>ALT5</b> — Select signal GPIO1_IO25.

### 30.4.67 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_SDSHR)

Address: 20E\_0000h base + 11Ch offset = 20E\_011Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_SDSHR field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_SDSHR. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: EPDC_SDSHR.  NOTE: Pad EPDC_SDSHR is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_SDSHR. 001 <b>ALT1</b> — Select signal ECSPI2_SCLK. - Configure register IOMUXC_ECSPI2_CSPI_CLK_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal EPDC_SDCE4. 011 <b>ALT3</b> — Select signal CSI_DATA11. - Configure register IOMUXC_CSI_CSI_DATA11_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SPDC_XDIOR. 101 <b>ALT5</b> — Select signal GPIO1_IO26.

### 30.4.68 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_VCOM0)

Address: 20E\_0000h base + 120h offset = 20E\_0120h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_VCOM0 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_VCOM0. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_VCOM0.  NOTE: Pad EPDC_VCOM0 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_VCOM0. 001 <b>ALT1</b> — Select signal AUD5_RXFS. - Configure register IOMUXC_AUD5_INPUT_RXFS_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal UART3_RX_DATA. - Configure register IOMUXC_UART3_UART_RX_DATA_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_ADDR24. 100 <b>ALT4</b> — Select signal SPDC_VCOM0. 101 <b>ALT5</b> — Select signal GPIO2_IO03. 110 <b>ALT6</b> — Select signal EPDC_SDCE5.

### 30.4.69 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_VCOM1)

Address: 20E\_0000h base + 124h offset = 20E\_0124h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_EPDC\_VCOM1 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad EPDC_VCOM1. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: EPDC_VCOM1.  NOTE: Pad EPDC_VCOM1 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal EPDC_VCOM1. 001 <b>ALT1</b> — Select signal AUD5_RXD. - Configure register IOMUXC_AUD5_INPUT_DA_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal UART3_TX_DATA. - Configure register IOMUXC_UART3_UART_RX_DATA_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_ADDR25. 100 <b>ALT4</b> — Select signal SPDC_VCOM1. 101 <b>ALT5</b> — Select signal GPIO2_IO04. 110 <b>ALT6</b> — Select signal EPDC_SDCE6.

### 30.4.70 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_CRSDV)

Address: 20E\_0000h base + 128h offset = 20E\_0128h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_CRSDV field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad FEC_CRSDV. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: FEC_CRSDV.  NOTE: Pad FEC_CRSDV is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal FEC_RX_DV. - Configure register IOMUXC_FEC_FEC_RX_DV_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal SD4_DATA1. - Configure register IOMUXC_USDHC4_DATA1_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal AUD6_TXC. - Configure register IOMUXC_AUD6_INPUT_TXCLK_AMX_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal ECSPI4_MISO. - Configure register IOMUXC_ECSPI4_MISO_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal GPT_COMPARE2. 101 <b>ALT5</b> — Select signal GPIO4_IO25. 110 <b>ALT6</b> — Select signal ARM_TRACE31.

### 30.4.71 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_MDC)

Address: 20E\_0000h base + 12Ch offset = 20E\_012Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_MDC field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad FEC_MDC. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: FEC_MDC.  NOTE: Pad FEC_MDC is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal FEC_MDC. 001 <b>ALT1</b> — Select signal SD4_DATA4. - Configure register IOMUXC_USDHC4_DATA4_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal AUDIO_CLK_OUT. 011 <b>ALT3</b> — Select signal SD1_RESET. 100 <b>ALT4</b> — Select signal SD3_RESET. 101 <b>ALT5</b> — Select signal GPIO4_IO23. 110 <b>ALT6</b> — Select signal ARM_TRACE29.

## 30.4.72 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_MDIO)

Address: 20E\_0000h base + 130h offset = 20E\_0130h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

### IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_MDIO field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad FEC_MDIO. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: FEC_MDIO.  NOTE: Pad FEC_MDIO is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal FEC_MDIO. - Configure register IOMUXC_FEC_FEC_MDI_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal SD4_CLK. - Configure register IOMUXC_USDHC4_CARD_CLK_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal AUD6_RXFS. - Configure register IOMUXC_AUD6_INPUT_RXFS_AMX_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal ECSPI4_SS0. - Configure register IOMUXC_ECSPI4_SS0_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal GPT_CAPTURE1. - Configure register IOMUXC_GPT_CAPIN1_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO20. 110 <b>ALT6</b> — Select signal ARM_TRACE26.



### 30.4.73 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_REF\_CLK)

Address: 20E\_0000h base + 134h offset = 20E\_0134h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_REF\_CLK field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad FEC_REF_CLK. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: FEC_REF_CLK.  NOTE: Pad FEC_REF_CLK is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal FEC_REF_OUT. 001 <b>ALT1</b> — Select signal SD4_RESET. 010 <b>ALT2</b> — Select signal WDOG1_B. 011 <b>ALT3</b> — Select signal PWM4_OUT. 100 <b>ALT4</b> — Select signal CCM_PMIC_READY. - Configure register IOMUXC_CCM_PMIC_READY_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO26. 110 <b>ALT6</b> — Select signal SPDIF_EXT_CLK. - Configure register IOMUXC_SPDIF_TX_CLK2_SELECT_INPUT for mode ALT6.

### 30.4.74 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_RX\_ER)

Address: 20E\_0000h base + 138h offset = 20E\_0138h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_RX\_ER field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad FEC_RX_ER. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: FEC_RX_ER.  NOTE: Pad FEC_RX_ER is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal FEC_RX_ER. - Configure register IOMUXC_FEC_FEC_RX_ER_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal SD4_DATA0. - Configure register IOMUXC_USDHC4_DATA0_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal AUD6_RXD. - Configure register IOMUXC_AUD6_INPUT_DA_AMX_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal ECSPI4_MOSI. - Configure register IOMUXC_ECSPi4_MOSI_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal GPT_COMPARE1. 101 <b>ALT5</b> — Select signal GPIO4_IO19. 110 <b>ALT6</b> — Select signal ARM_TRACE25.

### 30.4.75 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_RX\_DATA0)

Address: 20E\_0000h base + 13Ch offset = 20E\_013Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_RX\_DATA0 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad FEC_RXD0. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: FEC_RXD0.  NOTE: Pad FEC_RXD0 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal FEC_RX_DATA0. - Configure register IOMUXC_FEC_FEC_RX_DATA0_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal SD4_DATA5. - Configure register IOMUXC_USDHC4_DATA5_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal USB_OTG1_ID. - Configure register IOMUXC_ANALOG_USB_OTG_ID_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal SD1_VSELECT. 100 <b>ALT4</b> — Select signal SD3_VSELECT. 101 <b>ALT5</b> — Select signal GPIO4_IO17. 110 <b>ALT6</b> — Select signal ARM_TRACE24.

### 30.4.76 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_RX\_DATA1)

Address: 20E\_0000h base + 140h offset = 20E\_0140h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_RX\_DATA1 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad FEC_RXD1. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: FEC_RXD1.  NOTE: Pad FEC_RXD1 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal FEC_RX_DATA1. - Configure register IOMUXC_FEC_FEC_RX_DATA1_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal SD4_DATA2. - Configure register IOMUXC_USDHC4_DATA2_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal AUD6_TXFS. - Configure register IOMUXC_AUD6_INPUT_TXFS_AMX_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal ECSPI4_SS1. - Configure register IOMUXC_ECSP14_SS1_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal GPT_COMPARE3. 101 <b>ALT5</b> — Select signal GPIO4_IO18. 110 <b>ALT6</b> — Select signal FEC_COL. - Configure register IOMUXC_FEC_FEC_COL_SELECT_INPUT for mode ALT6.

### 30.4.77 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_TX\_CLK)

Address: 20E\_0000h base + 144h offset = 20E\_0144h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_TX\_CLK field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad FEC_TX_CLK. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: FEC_TX_CLK.  NOTE: Pad FEC_TX_CLK is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal FEC_TX_CLK. - Configure register IOMUXC_FEC_FEC_TX_CLK_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal SD4_CMD. - Configure register IOMUXC_USDHC4_CMD_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal AUD6_RXC. - Configure register IOMUXC_AUD6_INPUT_RXCLK_AMX_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal ECSPI4_SCLK. - Configure register IOMUXC_ECSPi4_CSPI_CLK_IN_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal GPT_CAPTURE2. - Configure register IOMUXC_GPT_CAPIN2_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO21. 110 <b>ALT6</b> — Select signal ARM_TRACE27.

### 30.4.78 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_TX\_EN)

Address: 20E\_0000h base + 148h offset = 20E\_0148h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_TX\_EN field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad FEC_TX_EN. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: FEC_TX_EN.  NOTE: Pad FEC_TX_EN is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal FEC_TX_EN. 001 <b>ALT1</b> — Select signal SD4_DATA6. - Configure register IOMUXC_USDHC4_DATA6_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal SPDIF_IN. - Configure register IOMUXC_SPDIF_SPDIF_IN1_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal SD1_WP. - Configure register IOMUXC_USDHC1_WP_ON_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SD3_WP. - Configure register IOMUXC_USDHC3_WP_ON_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO22. 110 <b>ALT6</b> — Select signal ARM_TRACE28.

### 30.4.79 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_TX\_DATA0)

Address: 20E\_0000h base + 14Ch offset = 20E\_014Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_TX\_DATA0 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad FEC_TXD0. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: FEC_TXD0.  NOTE: Pad FEC_TXD0 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal FEC_TX_DATA0. 001 <b>ALT1</b> — Select signal SD4_DATA3. - Configure register IOMUXC_USDHC4_DATA3_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal AUD6_TXD. - Configure register IOMUXC_AUD6_INPUT_DB_AMX_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal ECSPI4_SS2. - Configure register IOMUXC_ECSPi4_SS2_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal GPT_CLKIN. - Configure register IOMUXC_GPT_CLKIN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO24. 110 <b>ALT6</b> — Select signal ARM_TRACE30.

### 30.4.80 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_TX\_DATA1)

Address: 20E\_0000h base + 150h offset = 20E\_0150h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_FEC\_TX\_DATA1 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad FEC_TXD1. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: FEC_TXD1.  NOTE: Pad FEC_TXD1 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal FEC_TX_DATA1. 001 <b>ALT1</b> — Select signal SD4_DATA7. - Configure register IOMUXC_USDHC4_DATA7_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal SPDIF_OUT. 011 <b>ALT3</b> — Select signal SD1_CD_B. - Configure register IOMUXC_USDHC1_CARD_DET_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SD3_CD_B. - Configure register IOMUXC_USDHC3_CARD_DET_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO16. 110 <b>ALT6</b> — Select signal FEC_RX_CLK. - Configure register IOMUXC_FEC_FEC_RX_CLK_SELECT_INPUT for mode ALT6.



### 30.4.81 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_USB\_H\_DATA)

Address: 20E\_0000h base + 154h offset = 20E\_0154h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_USB\_H\_DATA field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad HSIC_DAT. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 5 iomux modes to be used for pad: HSIC_DAT.  NOTE: Pad HSIC_DAT is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal USB_H_DATA. 001 <b>ALT1</b> — Select signal I2C1_SCL. - Configure register IOMUXC_I2C1_SCL_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal PWM1_OUT. 011 <b>ALT3</b> — Select signal XTALOSC_REF_CLK_24M. 101 <b>ALT5</b> — Select signal GPIO3_IO19.

### 30.4.82 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_USB\_H\_STROBE)

Address: 20E\_0000h base + 158h offset = 20E\_0158h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_USB\_H\_STROBE field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad HSIC_STROBE. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 5 iomux modes to be used for pad: HSIC_STROBE.  NOTE: Pad HSIC_STROBE is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal USB_H_STROBE. 001 <b>ALT1</b> — Select signal I2C1_SDA. - Configure register IOMUXC_I2C1_SDA_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal PWM2_OUT. 011 <b>ALT3</b> — Select signal XTALOSC_REF_CLK_32K. 101 <b>ALT5</b> — Select signal GPIO3_IO20.

### 30.4.83 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_I2C1\_SCL)

Address: 20E\_0000h base + 15Ch offset = 20E\_015Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_I2C1\_SCL field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad I2C1_SCL. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: I2C1_SCL.  NOTE: Pad I2C1_SCL is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal I2C1_SCL. - Configure register IOMUXC_I2C1_SCL_IN_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal UART1_RTS_B. - Configure register IOMUXC_UART1_UART_RTS_B_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal ECSPI3_SS2. - Configure register IOMUXC_ECSPi3_SS2_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal FEC_RX_DATA0. - Configure register IOMUXC_FEC_FEC_RX_DATA0_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SD3_RESET. 101 <b>ALT5</b> — Select signal GPIO3_IO12. 110 <b>ALT6</b> — Select signal ECSPI1_SS1. - Configure register IOMUXC_ECSPi1_SS1_SELECT_INPUT for mode ALT6.

### 30.4.84 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_I2C1\_SDA)

Address: 20E\_0000h base + 160h offset = 20E\_0160h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_I2C1\_SDA field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad I2C1_SDA. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: I2C1_SDA.  NOTE: Pad I2C1_SDA is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal I2C1_SDA. - Configure register IOMUXC_I2C1_SDA_IN_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal UART1_CTS_B. - Configure register IOMUXC_UART1_UART_RTS_B_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal ECSPI3_SS3. - Configure register IOMUXC_ECSPI3_SS3_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal FEC_TX_EN. 100 <b>ALT4</b> — Select signal SD3_VSELECT. 101 <b>ALT5</b> — Select signal GPIO3_IO13. 110 <b>ALT6</b> — Select signal ECSPI1_SS2. - Configure register IOMUXC_ECSPI1_SS2_SELECT_INPUT for mode ALT6.

### 30.4.85 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_I2C2\_SCL)

Address: 20E\_0000h base + 164h offset = 20E\_0164h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_I2C2\_SCL field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad I2C2_SCL. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: I2C2_SCL.  NOTE: Pad I2C2_SCL is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal I2C2_SCL. - Configure register IOMUXC_I2C2_SCL_IN_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal AUD4_RXFS. - Configure register IOMUXC_AUD4_INPUT_RXFS_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal SPDIF_IN. - Configure register IOMUXC_SPDIF_SPDIF_IN1_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal FEC_TX_DATA1. 100 <b>ALT4</b> — Select signal SD3_WP. - Configure register IOMUXC_USDHC3_WP_ON_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO14. 110 <b>ALT6</b> — Select signal ECSPI1_RDY. - Configure register IOMUXC_ECSP11_DATAREADY_B_SELECT_INPUT for mode ALT6.

### 30.4.86 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_I2C2\_SDA)

Address: 20E\_0000h base + 168h offset = 20E\_0168h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_I2C2\_SDA field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad I2C2_SDA. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: I2C2_SDA.  NOTE: Pad I2C2_SDA is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal I2C2_SDA. - Configure register IOMUXC_I2C2_SDA_IN_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal AUD4_RXC. - Configure register IOMUXC_AUD4_INPUT_RXCLK_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal SPDIF_OUT. 011 <b>ALT3</b> — Select signal FEC_REF_OUT. 100 <b>ALT4</b> — Select signal SD3_CD_B. - Configure register IOMUXC_USDHC3_CARD_DET_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO15.

### 30.4.87 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_COL0)

Address: 20E\_0000h base + 16Ch offset = 20E\_016Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_COL0 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad KEY_COL0. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: KEY_COL0.  NOTE: Pad KEY_COL0 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal KEY_COL0. - Configure register IOMUXC_KEY_COL0_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal I2C2_SCL. - Configure register IOMUXC_I2C2_SCL_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA00. - Configure register IOMUXC_LCD_DATA00_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_AD00. 100 <b>ALT4</b> — Select signal SD1_CD_B. - Configure register IOMUXC_USDHC1_CARD_DET_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO24. 110 <b>ALT6</b> — Select signal MSHC_SCLK. - Configure register IOMUXC_MSHC_DI_SCKI_SELECT_INPUT for mode ALT6.

### 30.4.88 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_COL1)

Address: 20E\_0000h base + 170h offset = 20E\_0170h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_COL1 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad KEY_COL1. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: KEY_COL1.  NOTE: Pad KEY_COL1 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal KEY_COL1. - Configure register IOMUXC_KEY_COL1_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal ECSPI4_MOSI. - Configure register IOMUXC_ECSPI4_MOSI_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA02. - Configure register IOMUXC_LCD_DATA02_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_AD02. 100 <b>ALT4</b> — Select signal SD3_DATA4. - Configure register IOMUXC_USDHC3_DATA4_IN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO26. 110 <b>ALT6</b> — Select signal MSHC_DATA0. - Configure register IOMUXC_MSHC_DI_DATA0_SELECT_INPUT for mode ALT6.



### 30.4.89 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_COL2)

Address: 20E\_0000h base + 174h offset = 20E\_0174h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_COL2 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad KEY_COL2. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: KEY_COL2.  NOTE: Pad KEY_COL2 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal KEY_COL2. - Configure register IOMUXC_KEY_COL2_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal ECSPI4_SS0. - Configure register IOMUXC_ECSPi4_SS0_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA04. - Configure register IOMUXC_LCD_DATA04_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_AD04. 100 <b>ALT4</b> — Select signal SD3_DATA6. - Configure register IOMUXC_USDHC3_DATA6_IN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO28. 110 <b>ALT6</b> — Select signal MSHC_DATA2. - Configure register IOMUXC_MSHC_DI_DATA2_SELECT_INPUT for mode ALT6.

### 30.4.90 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_COL3)

Address: 20E\_0000h base + 178h offset = 20E\_0178h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_COL3 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad KEY_COL3. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: KEY_COL3.  NOTE: Pad KEY_COL3 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal KEY_COL3. - Configure register IOMUXC_KEY_COL3_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal AUD6_RXFS. - Configure register IOMUXC_AUD6_INPUT_RXFS_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA06. - Configure register IOMUXC_LCD_DATA06_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_AD06. 100 <b>ALT4</b> — Select signal SD4_DATA6. - Configure register IOMUXC_USDHC4_DATA6_IN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO30. 110 <b>ALT6</b> — Select signal SD1_RESET.

### 30.4.91 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_COL4)

Address: 20E\_0000h base + 17Ch offset = 20E\_017Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_COL4 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad KEY_COL4. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: KEY_COL4.  NOTE: Pad KEY_COL4 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal KEY_COL4. - Configure register IOMUXC_KEY_COL4_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal AUD6_RXD. - Configure register IOMUXC_AUD6_INPUT_DA_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA08. - Configure register IOMUXC_LCD_DATA08_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_AD08. 100 <b>ALT4</b> — Select signal SD4_CLK. - Configure register IOMUXC_USDHC4_CARD_CLK_IN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO00. 110 <b>ALT6</b> — Select signal USB_OTG1_PWR.

### 30.4.92 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_COL5)

Address: 20E\_0000h base + 180h offset = 20E\_0180h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_COL5 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad KEY_COL5. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: KEY_COL5.  NOTE: Pad KEY_COL5 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal KEY_COL5. - Configure register IOMUXC_KEY_COL5_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal AUD6_TXFS. - Configure register IOMUXC_AUD6_INPUT_TXFS_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA10. - Configure register IOMUXC_LCD_DATA10_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_AD10. 100 <b>ALT4</b> — Select signal SD4_DATA0. - Configure register IOMUXC_USDHC4_DATA0_IN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO02. 110 <b>ALT6</b> — Select signal USB_OTG2_PWR.

### 30.4.93 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_COL6)

Address: 20E\_0000h base + 184h offset = 20E\_0184h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_COL6 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad KEY_COL6. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: KEY_COL6.  NOTE: Pad KEY_COL6 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal KEY_COL6. - Configure register IOMUXC_KEY_COL6_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal UART4_RX_DATA. - Configure register IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA12. - Configure register IOMUXC_LCD_DATA12_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_AD12. 100 <b>ALT4</b> — Select signal SD4_DATA2. - Configure register IOMUXC_USDHC4_DATA2_IN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO04. 110 <b>ALT6</b> — Select signal SD3_RESET.

### 30.4.94 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_COL7)

Address: 20E\_0000h base + 188h offset = 20E\_0188h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_COL7 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad KEY_COL7. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: KEY_COL7.  NOTE: Pad KEY_COL7 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal KEY_COL7. - Configure register IOMUXC_KEY_COL7_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal UART4_RTS_B. - Configure register IOMUXC_UART4_UART_RTS_B_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA14. - Configure register IOMUXC_LCD_DATA14_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_AD14. 100 <b>ALT4</b> — Select signal SD4_DATA4. - Configure register IOMUXC_USDHC4_DATA4_IN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO06. 110 <b>ALT6</b> — Select signal SD1_WP. - Configure register IOMUXC_USDHC1_WP_ON_SELECT_INPUT for mode ALT6.

### 30.4.95 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_ROW0)

Address: 20E\_0000h base + 18Ch offset = 20E\_018Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_ROW0 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad KEY_ROW0. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: KEY_ROW0.  NOTE: Pad KEY_ROW0 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal KEY_ROW0. - Configure register IOMUXC_KEY_ROW0_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal I2C2_SDA. - Configure register IOMUXC_I2C2_SDA_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA01. - Configure register IOMUXC_LCD_DATA01_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_AD01. 100 <b>ALT4</b> — Select signal SD1_WP. - Configure register IOMUXC_USDHC1_WP_ON_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO25. 110 <b>ALT6</b> — Select signal MSHC_BS.

### 30.4.96 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_ROW1)

Address: 20E\_0000h base + 190h offset = 20E\_0190h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_ROW1 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad KEY_ROW1. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: KEY_ROW1.  NOTE: Pad KEY_ROW1 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal KEY_ROW1. - Configure register IOMUXC_KEY_ROW1_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal ECSPI4_MISO. - Configure register IOMUXC_ECSPi4_MISO_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA03. - Configure register IOMUXC_LCD_DATA03_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_AD03. 100 <b>ALT4</b> — Select signal SD3_DATA5. - Configure register IOMUXC_USDHC3_DATA5_IN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO27. 110 <b>ALT6</b> — Select signal MSHC_DATA1. - Configure register IOMUXC_MSHC_DI_DATA1_SELECT_INPUT for mode ALT6.



### 30.4.97 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_ROW2)

Address: 20E\_0000h base + 194h offset = 20E\_0194h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_ROW2 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad KEY_ROW2. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: KEY_ROW2.  NOTE: Pad KEY_ROW2 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal KEY_ROW2. - Configure register IOMUXC_KEY_ROW2_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal ECSPI4_SCLK. - Configure register IOMUXC_ECSPi4_CSPI_CLK_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA05. - Configure register IOMUXC_LCD_DATA05_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_AD05. 100 <b>ALT4</b> — Select signal SD3_DATA7. - Configure register IOMUXC_USDHC3_DATA7_IN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO29. 110 <b>ALT6</b> — Select signal MSHC_DATA3. - Configure register IOMUXC_MSHC_DI_DATA3_SELECT_INPUT for mode ALT6.

### 30.4.98 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_ROW3)

Address: 20E\_0000h base + 198h offset = 20E\_0198h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_ROW3 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad KEY_ROW3. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: KEY_ROW3.  NOTE: Pad KEY_ROW3 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal KEY_ROW3. - Configure register IOMUXC_KEY_ROW3_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal AUD6_RXC. - Configure register IOMUXC_AUD6_INPUT_RXCLK_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA07. - Configure register IOMUXC_LCD_DATA07_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_AD07. 100 <b>ALT4</b> — Select signal SD4_DATA7. - Configure register IOMUXC_USDHC4_DATA7_IN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO31. 110 <b>ALT6</b> — Select signal SD1_VSELECT.

### 30.4.99 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_ROW4)

Address: 20E\_0000h base + 19Ch offset = 20E\_019Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_ROW4 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad KEY_ROW4. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: KEY_ROW4.  NOTE: Pad KEY_ROW4 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal KEY_ROW4. - Configure register IOMUXC_KEY_ROW4_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal AUD6_TXC. - Configure register IOMUXC_AUD6_INPUT_TXCLK_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA09. - Configure register IOMUXC_LCD_DATA09_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_AD09. 100 <b>ALT4</b> — Select signal SD4_CMD. - Configure register IOMUXC_USDHC4_CMD_IN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO01. 110 <b>ALT6</b> — Select signal USB_OTG1_OC. - Configure register IOMUXC_USB_OTG1_OC_SELECT_INPUT for mode ALT6.

## 30.4.100 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_ROW5)

Address: 20E\_0000h base + 1A0h offset = 20E\_01A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

### IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_ROW5 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad KEY_ROW5. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: KEY_ROW5.  NOTE: Pad KEY_ROW5 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal KEY_ROW5. - Configure register IOMUXC_KEY_ROW5_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal AUD6_TXD. - Configure register IOMUXC_AUD6_INPUT_DB_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA11. - Configure register IOMUXC_LCD_DATA11_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_AD11. 100 <b>ALT4</b> — Select signal SD4_DATA1. - Configure register IOMUXC_USDHC4_DATA1_IN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO03. 110 <b>ALT6</b> — Select signal USB_OTG2_OC. - Configure register IOMUXC_USB_OTG2_OC_SELECT_INPUT for mode ALT6.

### 30.4.101 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_ROW6)

Address: 20E\_0000h base + 1A4h offset = 20E\_01A4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_ROW6 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad KEY_ROW6. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: KEY_ROW6.  NOTE: Pad KEY_ROW6 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal KEY_ROW6. - Configure register IOMUXC_KEY_ROW6_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal UART4_TX_DATA. - Configure register IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA13. - Configure register IOMUXC_LCD_DATA13_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_AD13. 100 <b>ALT4</b> — Select signal SD4_DATA3. - Configure register IOMUXC_USDHC4_DATA3_IN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO05. 110 <b>ALT6</b> — Select signal SD3_VSELECT.

### 30.4.102 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_ROW7)

Address: 20E\_0000h base + 1A8h offset = 20E\_01A8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_KEY\_ROW7 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad KEY_ROW7. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: KEY_ROW7.  NOTE: Pad KEY_ROW7 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal KEY_ROW7. - Configure register IOMUXC_KEY_ROW7_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal UART4_CTS_B. - Configure register IOMUXC_UART4_UART_RTS_B_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_DATA15. - Configure register IOMUXC_LCD_DATA15_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_AD15. 100 <b>ALT4</b> — Select signal SD4_DATA5. - Configure register IOMUXC_USDHC4_DATA5_IN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO07. 110 <b>ALT6</b> — Select signal SD1_CD_B. - Configure register IOMUXC_USDHC1_CARD_DET_SELECT_INPUT for mode ALT6.

### 30.4.103 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_CLK)

Address: 20E\_0000h base + 1ACh offset = 20E\_01ACh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_CLK field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_CLK. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: LCD_CLK.  NOTE: Pad LCD_CLK is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_CLK. 001 <b>ALT1</b> — Select signal SD4_DATA4. - Configure register IOMUXC_USDHC4_DATA4_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_WR_RWN. 011 <b>ALT3</b> — Select signal EIM_RW. 100 <b>ALT4</b> — Select signal PWM4_OUT. 101 <b>ALT5</b> — Select signal GPIO2_IO15.

### 30.4.104 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA0)

Address: 20E\_0000h base + 1B0h offset = 20E\_01B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA0 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DATA0. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DATA0.  NOTE: Pad LCD_DATA0 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA00. - Configure register IOMUXC_LCD_DATA00_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal ECSPI1_MOSI. - Configure register IOMUXC_ECSP11_MOSI_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal USB_OTG2_ID. - Configure register IOMUXC_ANALOG_USB_H1_ID_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal PWM1_OUT. 100 <b>ALT4</b> — Select signal UART5_DTR_B. 101 <b>ALT5</b> — Select signal GPIO2_IO20. 110 <b>ALT6</b> — Select signal ARM_TRACE00. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG00.



### 30.4.105 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA01)

Address: 20E\_0000h base + 1B4h offset = 20E\_01B4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA01 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT1. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT1.  NOTE: Pad LCD_DAT1 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA01. - Configure register IOMUXC_LCD_DATA01_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal ECSPI1_MISO. - Configure register IOMUXC_ECSP11_MISO_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal USB_OTG1_ID. - Configure register IOMUXC_ANALOG_USB_OTG_ID_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal PWM2_OUT. 100 <b>ALT4</b> — Select signal AUD4_RXFS. - Configure register IOMUXC_AUD4_INPUT_RXFS_AMX_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO2_IO21. 110 <b>ALT6</b> — Select signal ARM_TRACE01. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG01.

### 30.4.106 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA10)

Address: 20E\_0000h base + 1B8h offset = 20E\_01B8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA10 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT10. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT10.  NOTE: Pad LCD_DAT10 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA10. - Configure register IOMUXC_LCD_DATA10_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal KEY_COL1. - Configure register IOMUXC_KEY_COL1_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_DATA07. - Configure register IOMUXC_CSI_CSI_DATA07_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_DATA04. 100 <b>ALT4</b> — Select signal ECSPI2_MISO. - Configure register IOMUXC_ECSPi2_MISO_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO2_IO30. 110 <b>ALT6</b> — Select signal ARM_TRACE10. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG10.

### 30.4.107 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA11)

Address: 20E\_0000h base + 1BCh offset = 20E\_01BCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA11 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT11. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT11.  NOTE: Pad LCD_DAT11 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA11. - Configure register IOMUXC_LCD_DATA11_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal KEY_ROW1. - Configure register IOMUXC_KEY_ROW1_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_DATA06. - Configure register IOMUXC_CSI_CSI_DATA06_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_DATA05. 100 <b>ALT4</b> — Select signal ECSPi2_SS1. - Configure register IOMUXC_ECSPi2_SS1_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO2_IO31. 110 <b>ALT6</b> — Select signal ARM_TRACE11. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG11.

### 30.4.108 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA12)

Address: 20E\_0000h base + 1C0h offset = 20E\_01C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA12 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT12. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT12.  NOTE: Pad LCD_DAT12 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA12. - Configure register IOMUXC_LCD_DATA12_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal KEY_COL2. - Configure register IOMUXC_KEY_COL2_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_DATA05. - Configure register IOMUXC_CSI_CSI_DATA05_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_DATA06. 100 <b>ALT4</b> — Select signal UART5_RTS_B. - Configure register IOMUXC_UART5_UART_RTS_B_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO00. 110 <b>ALT6</b> — Select signal ARM_TRACE12. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG12.

### 30.4.109 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA13)

Address: 20E\_0000h base + 1C4h offset = 20E\_01C4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA13 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT13. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT13.  NOTE: Pad LCD_DAT13 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA13. - Configure register IOMUXC_LCD_DATA13_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal KEY_ROW2. - Configure register IOMUXC_KEY_ROW2_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_DATA04. - Configure register IOMUXC_CSI_CSI_DATA04_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_DATA07. 100 <b>ALT4</b> — Select signal UART5_CTS_B. - Configure register IOMUXC_UART5_UART_RTS_B_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO01. 110 <b>ALT6</b> — Select signal ARM_TRACE13. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG13.

### 30.4.110 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA14)

Address: 20E\_0000h base + 1C8h offset = 20E\_01C8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA14 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT14. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT14.  NOTE: Pad LCD_DAT14 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA14. - Configure register IOMUXC_LCD_DATA14_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal KEY_COL3. - Configure register IOMUXC_KEY_COL3_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_DATA03. - Configure register IOMUXC_CSI_CSI_DATA03_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_DATA08. 100 <b>ALT4</b> — Select signal UART5_RX_DATA. - Configure register IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO02. 110 <b>ALT6</b> — Select signal ARM_TRACE14. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG14.

### 30.4.111 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA15)

Address: 20E\_0000h base + 1CCh offset = 20E\_01CCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA15 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT15. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT15.  NOTE: Pad LCD_DAT15 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA15. - Configure register IOMUXC_LCD_DATA15_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal KEY_ROW3. - Configure register IOMUXC_KEY_ROW3_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_DATA02. - Configure register IOMUXC_CSI_CSI_DATA02_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_DATA09. 100 <b>ALT4</b> — Select signal UART5_TX_DATA. - Configure register IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO03. 110 <b>ALT6</b> — Select signal ARM_TRACE15. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG15.

### 30.4.112 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA16)

Address: 20E\_0000h base + 1D0h offset = 20E\_01D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA16 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT16. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT16.  NOTE: Pad LCD_DAT16 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA16. - Configure register IOMUXC_LCD_DATA16_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal KEY_COL4. - Configure register IOMUXC_KEY_COL4_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_DATA01. - Configure register IOMUXC_CSI_CSI_DATA01_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_DATA10. 100 <b>ALT4</b> — Select signal I2C2_SCL. - Configure register IOMUXC_I2C2_SCL_IN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO04. 110 <b>ALT6</b> — Select signal ARM_TRACE16. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG24.



### 30.4.113 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA17)

Address: 20E\_0000h base + 1D4h offset = 20E\_01D4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA17 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT17. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT17.  NOTE: Pad LCD_DAT17 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA17. - Configure register IOMUXC_LCD_DATA17_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal KEY_ROW4. - Configure register IOMUXC_KEY_ROW4_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_DATA00. - Configure register IOMUXC_CSI_CSI_DATA00_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_DATA11. 100 <b>ALT4</b> — Select signal I2C2_SDA. - Configure register IOMUXC_I2C2_SDA_IN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO05. 110 <b>ALT6</b> — Select signal ARM_TRACE17. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG25.

### 30.4.114 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA18)

Address: 20E\_0000h base + 1D8h offset = 20E\_01D8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA18 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT18. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT18.  NOTE: Pad LCD_DAT18 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA18. - Configure register IOMUXC_LCD_DATA18_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal KEY_COL5. - Configure register IOMUXC_KEY_COL5_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_DATA15. - Configure register IOMUXC_CSI_CSI_DATA15_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_DATA12. 100 <b>ALT4</b> — Select signal GPT_CAPTURE1. - Configure register IOMUXC_GPT_CAPIN1_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO06. 110 <b>ALT6</b> — Select signal ARM_TRACE18. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG26.

### 30.4.115 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA19)

Address: 20E\_0000h base + 1DCh offset = 20E\_01DCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA19 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT19. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT19.  NOTE: Pad LCD_DAT19 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA19. - Configure register IOMUXC_LCD_DATA19_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal KEY_ROW5. - Configure register IOMUXC_KEY_ROW5_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_DATA14. - Configure register IOMUXC_CSI_CSI_DATA14_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_DATA13. 100 <b>ALT4</b> — Select signal GPT_CAPTURE2. - Configure register IOMUXC_GPT_CAPIN2_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO07. 110 <b>ALT6</b> — Select signal ARM_TRACE19. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG27.

### 30.4.116 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA02)

Address: 20E\_0000h base + 1E0h offset = 20E\_01E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA02 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT2. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT2.  NOTE: Pad LCD_DAT2 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA02. - Configure register IOMUXC_LCD_DATA02_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal ECSPI1_SS0. - Configure register IOMUXC_ECSP11_SS0_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal EPIT2_OUT. 011 <b>ALT3</b> — Select signal PWM3_OUT. 100 <b>ALT4</b> — Select signal AUD4_RXC. - Configure register IOMUXC_AUD4_INPUT_RXCLK_AMX_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO2_IO22. 110 <b>ALT6</b> — Select signal ARM_TRACE02. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG02.

### 30.4.117 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA20)

Address: 20E\_0000h base + 1E4h offset = 20E\_01E4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA20 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT20. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT20.  NOTE: Pad LCD_DAT20 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA20. - Configure register IOMUXC_LCD_DATA20_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal KEY_COL6. - Configure register IOMUXC_KEY_COL6_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_DATA13. - Configure register IOMUXC_CSI_CSI_DATA13_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_DATA14. 100 <b>ALT4</b> — Select signal GPT_COMPARE1. 101 <b>ALT5</b> — Select signal GPIO3_IO08. 110 <b>ALT6</b> — Select signal ARM_TRACE20. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG28.

### 30.4.118 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA21)

Address: 20E\_0000h base + 1E8h offset = 20E\_01E8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA21 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT21. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT21.  NOTE: Pad LCD_DAT21 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA21. - Configure register IOMUXC_LCD_DATA21_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal KEY_ROW6. - Configure register IOMUXC_KEY_ROW6_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_DATA12. - Configure register IOMUXC_CSI_CSI_DATA12_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_DATA15. 100 <b>ALT4</b> — Select signal GPT_COMPARE2. 101 <b>ALT5</b> — Select signal GPIO3_IO09. 110 <b>ALT6</b> — Select signal ARM_TRACE21. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG29.

### 30.4.119 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA22)

Address: 20E\_0000h base + 1ECh offset = 20E\_01ECh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA22 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT22. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT22.  NOTE: Pad LCD_DAT22 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA22. - Configure register IOMUXC_LCD_DATA22_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal KEY_COL7. - Configure register IOMUXC_KEY_COL7_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_DATA11. - Configure register IOMUXC_CSI_CSI_DATA11_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_EB3_B. 100 <b>ALT4</b> — Select signal GPT_COMPARE3. 101 <b>ALT5</b> — Select signal GPIO3_IO10. 110 <b>ALT6</b> — Select signal ARM_TRACE22. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG30.

### 30.4.120 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA23)

Address: 20E\_0000h base + 1F0h offset = 20E\_01F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA23 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT23. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT23.  NOTE: Pad LCD_DAT23 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA23. - Configure register IOMUXC_LCD_DATA23_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal KEY_ROW7. - Configure register IOMUXC_KEY_ROW7_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_DATA10. - Configure register IOMUXC_CSI_CSI_DATA10_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_EB2_B. 100 <b>ALT4</b> — Select signal GPT_CLKIN. - Configure register IOMUXC_GPT_CLKIN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO11. 110 <b>ALT6</b> — Select signal ARM_TRACE23. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG31.



### 30.4.121 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA03)

Address: 20E\_0000h base + 1F4h offset = 20E\_01F4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA03 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT3. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT3.  NOTE: Pad LCD_DAT3 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA03. - Configure register IOMUXC_LCD_DATA03_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal ECSPI1_SCLK. - Configure register IOMUXC_ECSP11_CSPI_CLK_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal UART5_DSR_B. 011 <b>ALT3</b> — Select signal PWM4_OUT. 100 <b>ALT4</b> — Select signal AUD4_RXD. - Configure register IOMUXC_AUD4_INPUT_DA_AMX_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO2_IO23. 110 <b>ALT6</b> — Select signal ARM_TRACE03. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG03.

### 30.4.122 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA04)

Address: 20E\_0000h base + 1F8h offset = 20E\_01F8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA04 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DATA4. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DATA4.  NOTE: Pad LCD_DATA4 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA04. - Configure register IOMUXC_LCD_DATA04_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal ECSP11_SS1. - Configure register IOMUXC_ECSP11_SS1_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_VSYNC. - Configure register IOMUXC_CSI_CSI_VSYNC_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal WDOG2_RESET_B_DEB. 100 <b>ALT4</b> — Select signal AUD4_TXC. - Configure register IOMUXC_AUD4_INPUT_TXCLK_AMX_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO2_IO24. 110 <b>ALT6</b> — Select signal ARM_TRACE04. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG04.

### 30.4.123 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA05)

Address: 20E\_0000h base + 1FCh offset = 20E\_01FCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA05 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT5. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT5.  NOTE: Pad LCD_DAT5 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA05. - Configure register IOMUXC_LCD_DATA05_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal ECSP11_SS2. - Configure register IOMUXC_ECSP11_SS2_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_HSYNC. - Configure register IOMUXC_CSI_CSI_HSYNC_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_CS3_B. 100 <b>ALT4</b> — Select signal AUD4_TXFS. - Configure register IOMUXC_AUD4_INPUT_TXFS_AMX_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO2_IO25. 110 <b>ALT6</b> — Select signal ARM_TRACE05. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG05.

### 30.4.124 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA06)

Address: 20E\_0000h base + 200h offset = 20E\_0200h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA06 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT6. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT6.  NOTE: Pad LCD_DAT6 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA06. - Configure register IOMUXC_LCD_DATA06_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal ECSP11_SS3. - Configure register IOMUXC_ECSP11_SS3_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_PIXCLK. - Configure register IOMUXC_CSI_CSI_PIXCLK_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_DATA00. 100 <b>ALT4</b> — Select signal AUD4_TXD. - Configure register IOMUXC_AUD4_INPUT_DB_AMX_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO2_IO26. 110 <b>ALT6</b> — Select signal ARM_TRACE06. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG06.

### 30.4.125 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA07)

Address: 20E\_0000h base + 204h offset = 20E\_0204h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA07 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT7. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT7.  NOTE: Pad LCD_DAT7 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA07. - Configure register IOMUXC_LCD_DATA07_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal ECSPI1_RDY. - Configure register IOMUXC_ECSP11_DATAREADY_B_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_MCLK. 011 <b>ALT3</b> — Select signal EIM_DATA01. 100 <b>ALT4</b> — Select signal AUDIO_CLK_OUT. 101 <b>ALT5</b> — Select signal GPIO2_IO27. 110 <b>ALT6</b> — Select signal ARM_TRACE07. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG07.

### 30.4.126 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA08)

Address: 20E\_0000h base + 208h offset = 20E\_0208h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA08 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT8. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT8.  NOTE: Pad LCD_DAT8 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA08. - Configure register IOMUXC_LCD_DATA08_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal KEY_COL0. - Configure register IOMUXC_KEY_COL0_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_DATA09. - Configure register IOMUXC_CSI_CSI_DATA09_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_DATA02. 100 <b>ALT4</b> — Select signal ECSPi2_SCLK. - Configure register IOMUXC_ECSPi2_CSPI_CLK_IN_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO2_IO28. 110 <b>ALT6</b> — Select signal ARM_TRACE08. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG08.

### 30.4.127 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA09)

Address: 20E\_0000h base + 20Ch offset = 20E\_020Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_DATA09 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_DAT9. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 8 iomux modes to be used for pad: LCD_DAT9.  NOTE: Pad LCD_DAT9 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_DATA09. - Configure register IOMUXC_LCD_DATA09_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal KEY_ROW0. - Configure register IOMUXC_KEY_ROW0_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal CSI_DATA08. - Configure register IOMUXC_CSI_CSI_DATA08_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_DATA03. 100 <b>ALT4</b> — Select signal ECSPI2_MOSI. - Configure register IOMUXC_ECSPi2_MOSI_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO2_IO29. 110 <b>ALT6</b> — Select signal ARM_TRACE09. 111 <b>ALT7</b> — Select signal SRC_BOOT_CFG09.

### 30.4.128 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_ENABLE)

Address: 20E\_0000h base + 210h offset = 20E\_0210h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_ENABLE field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_ENABLE. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: LCD_ENABLE.  NOTE: Pad LCD_ENABLE is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_ENABLE. 001 <b>ALT1</b> — Select signal SD4_DATA5. - Configure register IOMUXC_USDHC4_DATA5_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_RD_E. 011 <b>ALT3</b> — Select signal EIM_OE_B. 100 <b>ALT4</b> — Select signal UART2_RX_DATA. - Configure register IOMUXC_UART2_UART_RX_DATA_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO2_IO16.



### 30.4.129 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_HSYNC)

Address: 20E\_0000h base + 214h offset = 20E\_0214h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_HSYNC field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_HSYNC. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: LCD_HSYNC.  NOTE: Pad LCD_HSYNC is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_HSYNC. - Configure register IOMUXC_LCD_BUSY_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal SD4_DATA6. - Configure register IOMUXC_USDHC4_DATA6_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_CS. 011 <b>ALT3</b> — Select signal EIM_CS0_B. 100 <b>ALT4</b> — Select signal UART2_TX_DATA. - Configure register IOMUXC_UART2_UART_RX_DATA_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO2_IO17. 110 <b>ALT6</b> — Select signal ARM_TRACE_CLK.

### 30.4.130 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_RESET)

Address: 20E\_0000h base + 218h offset = 20E\_0218h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_RESET field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_RESET. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: LCD_RESET.  NOTE: Pad LCD_RESET is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_RESET. 001 <b>ALT1</b> — Select signal EIM_DTACK_B. - Configure register IOMUXC_EIM_DTACK_B_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_BUSY. - Configure register IOMUXC_LCD_BUSY_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EIM_WAIT_B. - Configure register IOMUXC_EIM_WAIT_B_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal UART2_CTS_B. - Configure register IOMUXC_UART2_UART_RTS_B_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO2_IO19. 110 <b>ALT6</b> — Select signal CCM_PMIC_READY. - Configure register IOMUXC_CCM_PMIC_READY_SELECT_INPUT for mode ALT6.

### 30.4.131 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_VSYNC)

Address: 20E\_0000h base + 21Ch offset = 20E\_021Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_LCD\_VSYNC field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad LCD_VSYNC. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: LCD_VSYNC.  NOTE: Pad LCD_VSYNC is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal LCD_VSYNC. 001 <b>ALT1</b> — Select signal SD4_DATA7. - Configure register IOMUXC_USDHC4_DATA7_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal LCD_RS. 011 <b>ALT3</b> — Select signal EIM_CS1_B. 100 <b>ALT4</b> — Select signal UART2_RTS_B. - Configure register IOMUXC_UART2_UART_RTS_B_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO2_IO18. 110 <b>ALT6</b> — Select signal ARM_TRACE_CTL.

### 30.4.132 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_PWM1)

Address: 20E\_0000h base + 220h offset = 20E\_0220h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_PWM1 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad PWM1. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: PWM1.  000 <b>ALT0</b> — Select signal PWM1_OUT. 001 <b>ALT1</b> — Select signal CCM_CLKO. 010 <b>ALT2</b> — Select signal AUDIO_CLK_OUT. 011 <b>ALT3</b> — Select signal FEC_REF_OUT. 100 <b>ALT4</b> — Select signal CSI_MCLK. 101 <b>ALT5</b> — Select signal GPIO3_IO23. 110 <b>ALT6</b> — Select signal EPIT1_OUT.

### 30.4.133 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_REF\_CLK\_24M)

Address: 20E\_0000h base + 224h offset = 20E\_0224h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_REF\_CLK\_24M field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad REF_CLK_24M. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: REF_CLK_24M.  NOTE: Pad REF_CLK_24M is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal XTALOSC_REF_CLK_24M. 001 <b>ALT1</b> — Select signal I2C3_SCL. - Configure register IOMUXC_I2C3_SCL_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal PWM3_OUT. 011 <b>ALT3</b> — Select signal USB_OTG2_ID. - Configure register IOMUXC_ANALOG_USB_H1_ID_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal CCM_PMIC_READY. - Configure register IOMUXC_CCM_PMIC_READY_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO21. 110 <b>ALT6</b> — Select signal SD3_WP. - Configure register IOMUXC_USDHC3_WP_ON_SELECT_INPUT for mode ALT6.

### 30.4.134 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_REF\_CLK\_32K)

Address: 20E\_0000h base + 228h offset = 20E\_0228h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_REF\_CLK\_32K field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad REF_CLK_32K. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: REF_CLK_32K.  NOTE: Pad REF_CLK_32K is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal XTALOSC_REF_CLK_32K. 001 <b>ALT1</b> — Select signal I2C3_SDA. - Configure register IOMUXC_I2C3_SDA_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal PWM4_OUT. 011 <b>ALT3</b> — Select signal USB_OTG1_ID. - Configure register IOMUXC_ANALOG_USB_OTG_ID_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SD1_LCTL. 101 <b>ALT5</b> — Select signal GPIO3_IO22. 110 <b>ALT6</b> — Select signal SD3_CD_B. - Configure register IOMUXC_USDHC3_CARD_DET_SELECT_INPUT for mode ALT6.

### 30.4.135 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_CLK)

Address: 20E\_0000h base + 22Ch offset = 20E\_022Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_CLK field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD1_CLK. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 5 iomux modes to be used for pad: SD1_CLK.  NOTE: Pad SD1_CLK is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD1_CLK. 001 <b>ALT1</b> — Select signal FEC_MDIO. - Configure register IOMUXC_FEC_FEC_MDI_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal KEY_COL0. - Configure register IOMUXC_KEY_COL0_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EPDC_SDCE4. 100 <b>ALT4</b> — Select signal MSHC_SCLK. - Configure register IOMUXC_MSHC_DI_SCKI_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO5_IO15.

### 30.4.136 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_CMD)

Address: 20E\_0000h base + 230h offset = 20E\_0230h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_CMD field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD1_CMD. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 5 iomux modes to be used for pad: SD1_CMD.  NOTE: Pad SD1_CMD is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD1_CMD. 001 <b>ALT1</b> — Select signal FEC_TX_CLK. - Configure register IOMUXC_FEC_FEC_TX_CLK_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal KEY_ROW0. - Configure register IOMUXC_KEY_ROW0_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EPDC_SDCE5. 100 <b>ALT4</b> — Select signal MSHC_BS. 101 <b>ALT5</b> — Select signal GPIO5_IO14.



### 30.4.137 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_DATA0)

Address: 20E\_0000h base + 234h offset = 20E\_0234h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_DATA0 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD1_DATA0. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 5 iomux modes to be used for pad: SD1_DATA0.  NOTE: Pad SD1_DATA0 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD1_DATA0. 001 <b>ALT1</b> — Select signal FEC_RX_ER. - Configure register IOMUXC_FEC_FEC_RX_ER_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal KEY_COL1. - Configure register IOMUXC_KEY_COL1_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EPDC_SDCE6. 100 <b>ALT4</b> — Select signal MSHC_DATA0. - Configure register IOMUXC_MSHC_DI_DATA0_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO5_IO11.

### 30.4.138 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_DATA1)

Address: 20E\_0000h base + 238h offset = 20E\_0238h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_DATA1 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD1_DAT1. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 5 iomux modes to be used for pad: SD1_DAT1.  NOTE: Pad SD1_DAT1 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD1_DATA1. 001 <b>ALT1</b> — Select signal FEC_RX_DV. - Configure register IOMUXC_FEC_FEC_RX_DV_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal KEY_ROW1. - Configure register IOMUXC_KEY_ROW1_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EPDC_SDCE7. 100 <b>ALT4</b> — Select signal MSHC_DATA1. - Configure register IOMUXC_MSHC_DI_DATA1_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO5_IO08.

### 30.4.139 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_DATA2)

Address: 20E\_0000h base + 23Ch offset = 20E\_023Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_DATA2 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD1_DAT2. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 5 iomux modes to be used for pad: SD1_DAT2.  NOTE: Pad SD1_DAT2 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD1_DATA2. 001 <b>ALT1</b> — Select signal FEC_RX_DATA1. - Configure register IOMUXC_FEC_FEC_RX_DATA1_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal KEY_COL2. - Configure register IOMUXC_KEY_COL2_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EPDC_SDCE8. 100 <b>ALT4</b> — Select signal MSHC_DATA2. - Configure register IOMUXC_MSHC_DI_DATA2_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO5_IO13.

### 30.4.140 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_DATA3)

Address: 20E\_0000h base + 240h offset = 20E\_0240h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_DATA3 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD1_DAT3. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 5 iomux modes to be used for pad: SD1_DAT3.  NOTE: Pad SD1_DAT3 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD1_DATA3. 001 <b>ALT1</b> — Select signal FEC_TX_DATA0. 010 <b>ALT2</b> — Select signal KEY_ROW2. - Configure register IOMUXC_KEY_ROW2_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EPDC_SDCE9. 100 <b>ALT4</b> — Select signal MSHC_DATA3. - Configure register IOMUXC_MSHC_DI_DATA3_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO5_IO06.

### 30.4.141 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_DATA4)

Address: 20E\_0000h base + 244h offset = 20E\_0244h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_DATA4 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD1_DATA4. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: SD1_DATA4.  NOTE: Pad SD1_DATA4 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD1_DATA4. 001 <b>ALT1</b> — Select signal FEC_MDC. 010 <b>ALT2</b> — Select signal KEY_COL3. - Configure register IOMUXC_KEY_COL3_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EPDC_SDCLK_N. 100 <b>ALT4</b> — Select signal UART4_RX_DATA. - Configure register IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO5_IO12.

### 30.4.142 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_DATA5)

Address: 20E\_0000h base + 248h offset = 20E\_0248h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_DATA5 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD1_DAT5. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: SD1_DAT5.  NOTE: Pad SD1_DAT5 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD1_DATA5. 001 <b>ALT1</b> — Select signal FEC_RX_DATA0. - Configure register IOMUXC_FEC_FEC_RX_DATA0_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal KEY_ROW3. - Configure register IOMUXC_KEY_ROW3_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EPDC_SDOED. 100 <b>ALT4</b> — Select signal UART4_TX_DATA. - Configure register IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO5_IO09.

### 30.4.143 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_DATA6)

Address: 20E\_0000h base + 24Ch offset = 20E\_024Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_DATA6 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD1_DATA6. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: SD1_DATA6.  NOTE: Pad SD1_DATA6 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD1_DATA6. 001 <b>ALT1</b> — Select signal FEC_TX_EN. 010 <b>ALT2</b> — Select signal KEY_COL4. - Configure register IOMUXC_KEY_COL4_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal EPDC_SDOEZ. 100 <b>ALT4</b> — Select signal UART4_RTS_B. - Configure register IOMUXC_UART4_UART_RTS_B_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO5_IO07.

### 30.4.144 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_DATA7)

Address: 20E\_0000h base + 250h offset = 20E\_0250h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD1\_DATA7 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD1_DAT7. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: SD1_DAT7.  NOTE: Pad SD1_DAT7 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD1_DATA7. 001 <b>ALT1</b> — Select signal FEC_TX_DATA1. 010 <b>ALT2</b> — Select signal KEY_ROW4. - Configure register IOMUXC_KEY_ROW4_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CCM_PMIC_READY. - Configure register IOMUXC_CCM_PMIC_READY_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal UART4_CTS_B. - Configure register IOMUXC_UART4_UART_RTS_B_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO5_IO10.



### 30.4.145 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_CLK)

Address: 20E\_0000h base + 254h offset = 20E\_0254h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_CLK field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD2_CLK. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 5 iomux modes to be used for pad: SD2_CLK.  NOTE: Pad SD2_CLK is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD2_CLK. 001 <b>ALT1</b> — Select signal AUD4_RXFS. - Configure register IOMUXC_AUD4_INPUT_RXFS_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal ECSPI3_SCLK. - Configure register IOMUXC_ECSPi3_CSPI_CLK_IN_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA00. - Configure register IOMUXC_CSI_CSI_DATA00_SELECT_INPUT for mode ALT3. 101 <b>ALT5</b> — Select signal GPIO5_IO05.

### 30.4.146 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_CMD)

Address: 20E\_0000h base + 258h offset = 20E\_0258h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_CMD field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD2_CMD. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: SD2_CMD.  NOTE: Pad SD2_CMD is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD2_CMD. 001 <b>ALT1</b> — Select signal AUD4_RXC. - Configure register IOMUXC_AUD4_INPUT_RXCLK_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal ECSPI3_SS0. - Configure register IOMUXC_ECSPI3_SS0_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA01. - Configure register IOMUXC_CSI_CSI_DATA01_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal EPIT1_OUT. 101 <b>ALT5</b> — Select signal GPIO5_IO04.

### 30.4.147 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_DATA0)

Address: 20E\_0000h base + 25Ch offset = 20E\_025Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_DATA0 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD2_DATA0. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: SD2_DATA0.  NOTE: Pad SD2_DATA0 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD2_DATA0. 001 <b>ALT1</b> — Select signal AUD4_RXD. - Configure register IOMUXC_AUD4_INPUT_DA_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal ECSPI3_MOSI. - Configure register IOMUXC_ECSPI3_MOSI_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA02. - Configure register IOMUXC_CSI_CSI_DATA02_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal UART5_RTS_B. - Configure register IOMUXC_UART5_UART_RTS_B_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO5_IO01.

### 30.4.148 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_DATA1)

Address: 20E\_0000h base + 260h offset = 20E\_0260h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_DATA1 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD2_DAT1. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: SD2_DAT1.  NOTE: Pad SD2_DAT1 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD2_DATA1. 001 <b>ALT1</b> — Select signal AUD4_TXC. - Configure register IOMUXC_AUD4_INPUT_TXCLK_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal ECSPI3_MISO. - Configure register IOMUXC_ECSPI3_MISO_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA03. - Configure register IOMUXC_CSI_CSI_DATA03_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal UART5_CTS_B. - Configure register IOMUXC_UART5_UART_RTS_B_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO30.

### 30.4.149 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_DATA2)

Address: 20E\_0000h base + 264h offset = 20E\_0264h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_DATA2 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD2_DATA2. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: SD2_DATA2.  NOTE: Pad SD2_DATA2 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD2_DATA2. 001 <b>ALT1</b> — Select signal AUD4_TXFS. - Configure register IOMUXC_AUD4_INPUT_TXFS_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal FEC_COL. - Configure register IOMUXC_FEC_FEC_COL_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA04. - Configure register IOMUXC_CSI_CSI_DATA04_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal UART5_RX_DATA. - Configure register IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO5_IO03.

### 30.4.150 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_DATA3)

Address: 20E\_0000h base + 268h offset = 20E\_0268h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_DATA3 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD2_DAT3. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: SD2_DAT3.  NOTE: Pad SD2_DAT3 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD2_DATA3. 001 <b>ALT1</b> — Select signal AUD4_TXD. - Configure register IOMUXC_AUD4_INPUT_DB_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal FEC_RX_CLK. - Configure register IOMUXC_FEC_FEC_RX_CLK_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA05. - Configure register IOMUXC_CSI_CSI_DATA05_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal UART5_TX_DATA. - Configure register IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO28.

### 30.4.151 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_DATA4)

Address: 20E\_0000h base + 26Ch offset = 20E\_026Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_DATA4 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD2_DATA4. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: SD2_DATA4.  NOTE: Pad SD2_DATA4 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD2_DATA4. 001 <b>ALT1</b> — Select signal SD3_DATA4. - Configure register IOMUXC_USDHC3_DATA4_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal UART2_RX_DATA. - Configure register IOMUXC_UART2_UART_RX_DATA_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA06. - Configure register IOMUXC_CSI_CSI_DATA06_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SPDIF_OUT. 101 <b>ALT5</b> — Select signal GPIO5_IO02.

## 30.4.152 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_DATA5)

Address: 20E\_0000h base + 270h offset = 20E\_0270h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_DATA5 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD2_DAT5. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: SD2_DAT5.  NOTE: Pad SD2_DAT5 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD2_DATA5. 001 <b>ALT1</b> — Select signal SD3_DATA5. - Configure register IOMUXC_USDHC3_DATA5_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal UART2_TX_DATA. - Configure register IOMUXC_UART2_UART_RX_DATA_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA07. - Configure register IOMUXC_CSI_CSI_DATA07_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SPDIF_IN. - Configure register IOMUXC_SPDIF_SPDIF_IN1_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO31.



### 30.4.153 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_DATA6)

Address: 20E\_0000h base + 274h offset = 20E\_0274h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_DATA6 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD2_DAT6. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: SD2_DAT6.  NOTE: Pad SD2_DAT6 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD2_DATA6. 001 <b>ALT1</b> — Select signal SD3_DATA6. - Configure register IOMUXC_USDHC3_DATA6_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal UART2_RTS_B. - Configure register IOMUXC_UART2_UART_RTS_B_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA08. - Configure register IOMUXC_CSI_CSI_DATA08_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SD2_WP. - Configure register IOMUXC_USDHC2_WP_ON_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO4_IO29.

### 30.4.154 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_DATA7)

Address: 20E\_0000h base + 278h offset = 20E\_0278h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_DATA7 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD2_DAT7. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: SD2_DAT7.  NOTE: Pad SD2_DAT7 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD2_DATA7. 001 <b>ALT1</b> — Select signal SD3_DATA7. - Configure register IOMUXC_USDHC3_DATA7_IN_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal UART2_CTS_B. - Configure register IOMUXC_UART2_UART_RTS_B_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA09. - Configure register IOMUXC_CSI_CSI_DATA09_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SD2_CD_B. - Configure register IOMUXC_USDHC2_CARD_DET_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO5_IO00.

### 30.4.155 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_RESET)

Address: 20E\_0000h base + 27Ch offset = 20E\_027Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W													SION			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD2\_RESET field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD2_RST. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: SD2_RST.  000 <b>ALT0</b> — Select signal SD2_RESET. 001 <b>ALT1</b> — Select signal FEC_REF_OUT. 010 <b>ALT2</b> — Select signal WDOG2_B. 011 <b>ALT3</b> — Select signal SPDIF_OUT. 100 <b>ALT4</b> — Select signal CSI_MCLK. 101 <b>ALT5</b> — Select signal GPIO4_IO27.

### 30.4.156 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD3\_CLK)

Address: 20E\_0000h base + 280h offset = 20E\_0280h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD3\_CLK field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD3_CLK. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: SD3_CLK.  NOTE: Pad SD3_CLK is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD3_CLK. 001 <b>ALT1</b> — Select signal AUD5_RXFS. - Configure register IOMUXC_AUD5_INPUT_RXFS_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal KEY_COL5. - Configure register IOMUXC_KEY_COL5_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA10. - Configure register IOMUXC_CSI_CSI_DATA10_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal WDOG1_RESET_B_DEB. 101 <b>ALT5</b> — Select signal GPIO5_IO18. 110 <b>ALT6</b> — Select signal USB_OTG1_PWR.

### 30.4.157 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD3\_CMD)

Address: 20E\_0000h base + 284h offset = 20E\_0284h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD3\_CMD field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD3_CMD. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: SD3_CMD.  NOTE: Pad SD3_CMD is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD3_CMD. 001 <b>ALT1</b> — Select signal AUD5_RXC. - Configure register IOMUXC_AUD5_INPUT_RXCLK_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal KEY_ROW5. - Configure register IOMUXC_KEY_ROW5_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA11. - Configure register IOMUXC_CSI_CSI_DATA11_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal USB_OTG2_ID. - Configure register IOMUXC_ANALOG_USB_H1_ID_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO5_IO21. 110 <b>ALT6</b> — Select signal USB_OTG2_PWR.

### 30.4.158 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD3\_DATA0)

Address: 20E\_0000h base + 288h offset = 20E\_0288h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD3\_DATA0 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD3_DATA0. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: SD3_DATA0.  NOTE: Pad SD3_DATA0 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD3_DATA0. 001 <b>ALT1</b> — Select signal AUD5_RXD. - Configure register IOMUXC_AUD5_INPUT_DA_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal KEY_COL6. - Configure register IOMUXC_KEY_COL6_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA12. - Configure register IOMUXC_CSI_CSI_DATA12_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal USB_OTG1_ID. - Configure register IOMUXC_ANALOG_USB_OTG_ID_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO5_IO19.

### 30.4.159 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD3\_DATA1)

Address: 20E\_0000h base + 28Ch offset = 20E\_028Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD3\_DATA1 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD3_DAT1. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: SD3_DAT1.  NOTE: Pad SD3_DAT1 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD3_DATA1. 001 <b>ALT1</b> — Select signal AUD5_TXC. - Configure register IOMUXC_AUD5_INPUT_TXCLK_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal KEY_ROW6. - Configure register IOMUXC_KEY_ROW6_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA13. - Configure register IOMUXC_CSI_CSI_DATA13_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal SD1_VSELECT. 101 <b>ALT5</b> — Select signal GPIO5_IO20. 110 <b>ALT6</b> — Select signal JTAG_DE_B.

### 30.4.160 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD3\_DATA2)

Address: 20E\_0000h base + 290h offset = 20E\_0290h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD3\_DATA2 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD3_DATA2. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: SD3_DATA2.  NOTE: Pad SD3_DATA2 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD3_DATA2. 001 <b>ALT1</b> — Select signal AUD5_TXFS. - Configure register IOMUXC_AUD5_INPUT_TXFS_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal KEY_COL7. - Configure register IOMUXC_KEY_COL7_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA14. - Configure register IOMUXC_CSI_CSI_DATA14_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal EPIT1_OUT. 101 <b>ALT5</b> — Select signal GPIO5_IO16. 110 <b>ALT6</b> — Select signal USB_OTG2_OC. - Configure register IOMUXC_USB_OTG2_OC_SELECT_INPUT for mode ALT6.



### 30.4.161 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_SD3\_DATA3)

Address: 20E\_0000h base + 294h offset = 20E\_0294h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0	MUX_MODE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_SD3\_DATA3 field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad SD3_DAT3. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: SD3_DAT3.  NOTE: Pad SD3_DAT3 is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal SD3_DATA3. 001 <b>ALT1</b> — Select signal AUD5_TXD. - Configure register IOMUXC_AUD5_INPUT_DB_AMX_SELECT_INPUT for mode ALT1. 010 <b>ALT2</b> — Select signal KEY_ROW7. - Configure register IOMUXC_KEY_ROW7_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal CSI_DATA15. - Configure register IOMUXC_CSI_CSI_DATA15_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal EPIT2_OUT. 101 <b>ALT5</b> — Select signal GPIO5_IO17. 110 <b>ALT6</b> — Select signal USB_OTG1_OC. - Configure register IOMUXC_USB_OTG1_OC_SELECT_INPUT for mode ALT6.

## 30.4.162 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_UART1\_RXD)

Address: 20E\_0000h base + 298h offset = 20E\_0298h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

### IOMUXC\_SW\_MUX\_CTL\_PAD\_UART1\_RXD field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad UART1_RXD. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 6 iomux modes to be used for pad: UART1_RXD.  NOTE: Pad UART1_RXD is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal UART1_RX_DATA. - Configure register IOMUXC_UART1_UART_RX_DATA_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal PWM1_OUT. 010 <b>ALT2</b> — Select signal UART4_RX_DATA. - Configure register IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal FEC_COL. - Configure register IOMUXC_FEC_FEC_COL_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal UART5_RX_DATA. - Configure register IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO16.

### 30.4.163 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_UART1\_TXD)

Address: 20E\_0000h base + 29Ch offset = 20E\_029Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_UART1\_TXD field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad UART1_TXD. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 7 iomux modes to be used for pad: UART1_TXD.  NOTE: Pad UART1_TXD is involved in Daisy Chain.  000 <b>ALT0</b> — Select signal UART1_TX_DATA. - Configure register IOMUXC_UART1_UART_RX_DATA_SELECT_INPUT for mode ALT0. 001 <b>ALT1</b> — Select signal PWM2_OUT. 010 <b>ALT2</b> — Select signal UART4_TX_DATA. - Configure register IOMUXC_UART4_UART_RX_DATA_SELECT_INPUT for mode ALT2. 011 <b>ALT3</b> — Select signal FEC_RX_CLK. - Configure register IOMUXC_FEC_FEC_RX_CLK_SELECT_INPUT for mode ALT3. 100 <b>ALT4</b> — Select signal UART5_TX_DATA. - Configure register IOMUXC_UART5_UART_RX_DATA_SELECT_INPUT for mode ALT4. 101 <b>ALT5</b> — Select signal GPIO3_IO17. 111 <b>ALT7</b> — Select signal UART5_DCD_B.

### 30.4.164 Pad Mux Register (IOMUXC\_SW\_MUX\_CTL\_PAD\_WDOG\_B)

Address: 20E\_0000h base + 2A0h offset = 20E\_02A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												SION	0	MUX_MODE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

#### IOMUXC\_SW\_MUX\_CTL\_PAD\_WDOG\_B field descriptions

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 SION	Software Input On Field.  Force the selected mux mode input path no matter of MUX_MODE functionality.  1 <b>ENABLED</b> — Force input path of pad WDOG_B. 0 <b>DISABLED</b> — Input Path is determined by functionality of the selected mux mode (regular).
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 MUX_MODE	MUX Mode Select Field.  Select 1 of 4 iomux modes to be used for pad: WDOG_B.  000 <b>ALT0</b> — Select signal WDOG1_B. 001 <b>ALT1</b> — Select signal WDOG1_RESET_B_DEB. 010 <b>ALT2</b> — Select signal UART5_RI_B. 101 <b>ALT5</b> — Select signal GPIO3_IO18.

### 30.4.165 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_MCLK)

Address: 20E\_0000h base + 2A4h offset = 20E\_02A4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_MCLK field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: AUD_MCLK.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: AUD_MCLK.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: AUD_MCLK.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: AUD_MCLK.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: AUD_MCLK.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_MCLK field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: AUD_MCLK.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: AUD_MCLK.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.166 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_RXC)

Address: 20E\_0000h base + 2A8h offset = 20E\_02A8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE	0					HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS	PUE	PKE	ODE	0				SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_RXC field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: AUD_RXC.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: AUD_RXC.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: AUD_RXC.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: AUD_RXC.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: AUD_RXC.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_RXC field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: AUD_RXC.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: AUD_RXC.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.167 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_RXD)

Address: 20E\_0000h base + 2ACh offset = 20E\_02ACh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE	0					HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS	PUE	PKE	ODE	0				SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_RXD field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: AUD_RXD.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: AUD_RXD.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: AUD_RXD.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: AUD_RXD.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: AUD_RXD.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_RXD field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: AUD_RXD.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: AUD_RXD.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.168 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_RXFS)

Address: 20E\_0000h base + 2B0h offset = 20E\_02B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_RXFS field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: AUD_RXFS.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: AUD_RXFS.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: AUD_RXFS.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: AUD_RXFS.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: AUD_RXFS.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_RXFS field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: AUD_RXFS.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: AUD_RXFS.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.169 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_TXC)

Address: 20E\_0000h base + 2B4h offset = 20E\_02B4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		
W																SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_TXC field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: AUD_TXC.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: AUD_TXC.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: AUD_TXC.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: AUD_TXC.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: AUD_TXC.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_TXC field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: AUD_TXC.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: AUD_TXC.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.170 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_TXD)

Address: 20E\_0000h base + 2B8h offset = 20E\_02B8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_TXD field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: AUD_TXD.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: AUD_TXD.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: AUD_TXD.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: AUD_TXD.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: AUD_TXD.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_TXD field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: AUD_TXD.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: AUD_TXD.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.171 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_TXFS)

Address: 20E\_0000h base + 2BCh offset = 20E\_02BCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_TXFS field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: AUD_TXFS.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: AUD_TXFS.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: AUD_TXFS.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: AUD_TXFS.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: AUD_TXFS.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_AUD\_TXFS field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: AUD_TXFS. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: AUD_TXFS. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.172 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR00)

Address: 20E\_0000h base + 2C0h offset = 20E\_02C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR00 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_A0.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_A0.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR00 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_A0.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.173 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR01)

Address: 20E\_0000h base + 2C4h offset = 20E\_02C4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR01 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_A1.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_A1.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR01 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_A1.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.174 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR10)

Address: 20E\_0000h base + 2C8h offset = 20E\_02C8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0	DSE			0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR10 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_A10.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_A10.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR10 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_A10.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.



### 30.4.175 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR11)

Address: 20E\_0000h base + 2CCh offset = 20E\_02CCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR11 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_A11.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_A11.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR11 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_A11.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.176 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR12)

Address: 20E\_0000h base + 2D0h offset = 20E\_02D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR12 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_A12.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_A12.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR12 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_A12.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.177 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR13)

Address: 20E\_0000h base + 2D4h offset = 20E\_02D4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR13 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_A13.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_A13.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR13 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_A13.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.178 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR14)

Address: 20E\_0000h base + 2D8h offset = 20E\_02D8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	0	ODT		0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR14 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_A14.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_A14.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR14 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_A14.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.



### 30.4.179 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR15)

Address: 20E\_0000h base + 2DCh offset = 20E\_02DCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR15 field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_A15.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_A15.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR15 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_A15.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.180 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR02)

Address: 20E\_0000h base + 2E0h offset = 20E\_02E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR02 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_A2.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_A2.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR02 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_A2.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.181 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR03)

Address: 20E\_0000h base + 2E4h offset = 20E\_02E4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR03 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_A3.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_A3.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR03 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_A3.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.182 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR04)

Address: 20E\_0000h base + 2E8h offset = 20E\_02E8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR04 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_A4.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_A4.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR04 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_A4.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.



### 30.4.183 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR05)

Address: 20E\_0000h base + 2ECh offset = 20E\_02ECh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR05 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_A5.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_A5.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR05 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_A5.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.184 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR06)

Address: 20E\_0000h base + 2F0h offset = 20E\_02F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR06 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_A6.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_A6.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR06 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_A6.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.185 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR07)

Address: 20E\_0000h base + 2F4h offset = 20E\_02F4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	0	ODT		0	DSE			0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR07 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_A7.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_A7.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR07 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_A7.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.186 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR08)

Address: 20E\_0000h base + 2F8h offset = 20E\_02F8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	0	ODT		0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR08 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_A8.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_A8.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR08 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_A8.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.



### 30.4.187 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR09)

Address: 20E\_0000h base + 2FCh offset = 20E\_02FCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR09 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_A9.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_A9.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ADDR09 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_A9.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.188 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_CAS\_B)

Address: 20E\_0000h base + 300h offset = 20E\_0300h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	0	ODT		0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_CAS\_B field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_CAS.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_CAS.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_CAS\_B field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_CAS.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  Select one of next values for pad: DRAM_CAS.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.189 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_CS0\_B)

Address: 20E\_0000h base + 304h offset = 20E\_0304h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0	DSE			0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_CS0\_B field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_CS0.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_CS0.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_CS0\_B field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_CS0.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_CTLDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.190 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_CS1\_B)

Address: 20E\_0000h base + 308h offset = 20E\_0308h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_CS1\_B field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_CS1.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_CS1.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_CS1\_B field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_CS1.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_CTLDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.



### 30.4.191 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_DQM0)

Address: 20E\_0000h base + 30Ch offset = 20E\_030Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	0	ODT		0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_DQM0 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_DQM0.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_DQM0.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_DQM0 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_DQM0.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  Select one of next values for pad: DRAM_DQM0.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.192 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_DQM1)

Address: 20E\_0000h base + 310h offset = 20E\_0310h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0		DSE			0		
W																
Reset	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_DQM1 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_DQM1.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_DQM1.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_DQM1 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_DQM1.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  Select one of next values for pad: DRAM_DQM1.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.193 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_DQM2)

Address: 20E\_0000h base + 314h offset = 20E\_0314h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	0	ODT		0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_DQM2 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_DQM2.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_DQM2.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_DQM2 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_DQM2.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  Select one of next values for pad: DRAM_DQM2.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.194 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_DQM3)

Address: 20E\_0000h base + 318h offset = 20E\_0318h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	0	ODT		0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_DQM3 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_DQM3.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_DQM3.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_DQM3 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_DQM3.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  Select one of next values for pad: DRAM_DQM3.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–0 Reserved	This read-only field is reserved and always has the value 0.



### 30.4.195 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_RAS\_B)

Address: 20E\_0000h base + 31Ch offset = 20E\_031Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	0	ODT		0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_RAS\_B field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_RAS.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_RAS.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_RAS\_B field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_RAS.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  Select one of next values for pad: DRAM_RAS.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.196 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_RESET)

Address: 20E\_0000h base + 320h offset = 20E\_0320h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0	DSE			0			
W																
Reset	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_RESET field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	<p>DDR Select Field</p> <p>Select one of next values for pad: DRAM_RESET.</p> <p>00 <b>RESERVED0</b> — Reserved</p> <p>01 <b>RESERVED1</b> — Reserved</p> <p>10 <b>LPDDR2</b> — LPDDR2 mode (240 Ohm driver unit calibration, 240, 120, 80, 60, 48, 40, 32 Ohm drive strengths at 1.2V)</p> <p>11 <b>DDR3</b> — DDR3 mode (240 Ohm driver unit calibration, 240, 120, 80, 60, 48, 40, 32 Ohm drive strengths at 1.5V)</p>
17 DDR_INPUT	<p>DDR / CMOS Input Mode Field</p> <p>Select one of next values for pad: DRAM_RESET.</p> <p>0 <b>CMOS</b> — CMOS input mode.</p> <p>1 <b>DIFFERENTIAL</b> — Differential input mode.</p>
16 HYS	<p>Hysteresis Enable Field</p> <p>Select one of next values for pad: DRAM_RESET.</p> <p>0 <b>DISABLED</b> — CMOS input</p> <p>1 <b>ENABLED</b> — Schmitt trigger input</p>
15–14 PUS	<p>Pull Up / Down Config. Field</p> <p>Select one of next values for pad: DRAM_RESET.</p>

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_RESET field descriptions (continued)**

Field	Description
	00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: DRAM_RESET. 0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: DRAM_RESET. 0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field Select one of next values for pad: DRAM_RESET. 000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field Select one of next values for pad: DRAM_RESET. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.197 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDBA0)

Address: 20E\_0000h base + 324h offset = 20E\_0324h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0	DSE			0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDBA0 field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_SDBA0.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_SDBA0.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDBA0 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_SDBA0.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.198 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDBA1)

Address: 20E\_0000h base + 328h offset = 20E\_0328h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0		DSE		0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDBA1 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_SDBA1.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_SDBA1.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDBA1 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_SDBA1.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_ADDDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.



### 30.4.199 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDBA2)

Address: 20E\_0000h base + 32Ch offset = 20E\_032Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0	DSE			0			
W																
Reset	1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDBA2 field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_SDBA2.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_SDBA2.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDBA2 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  Select one of next values for pad: DRAM_SDBA2.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field  Select one of next values for pad: DRAM_SDBA2.  0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_SDBA2.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_CTLDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.200 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDCKE0)

Address: 20E\_0000h base + 330h offset = 20E\_0330h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0	DSE		0				
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDCKE0 field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_SDCKE0.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_SDCKE0.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDCKE0 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Select one of next values for pad: DRAM_SDCKE0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field  Select one of next values for pad: DRAM_SDCKE0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field  Select one of next values for pad: DRAM_SDCKE0.  0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_SDCKE0.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_CTLDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.201 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDCKE1)

Address: 20E\_0000h base + 334h offset = 20E\_0334h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0	DSE		0				
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDCKE1 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_SDCKE1.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_SDCKE1.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDCKE1 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: DRAM_SDCKE1.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: DRAM_SDCKE1.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: DRAM_SDCKE1.  0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field Select one of next values for pad: DRAM_SDCKE1.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_CTLDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.202 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDCLK0\_P)

Address: 20E\_0000h base + 338h offset = 20E\_0338h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0		DSE			0		
W																
Reset	1	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDCLK0\_P field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_SDCLK_0.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_SDCLK_0.

*Table continues on the next page...*

## IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDCLK0\_P field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_SDCLK_0.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  Select one of next values for pad: DRAM_SDCLK_0.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–0 Reserved	This read-only field is reserved and always has the value 0.



### 30.4.203 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ODT0)

Address: 20E\_0000h base + 33Ch offset = 20E\_033Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0		DSE		0			
W																
Reset	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ODT0 field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_SDODT0.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_SDODT0.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ODT0 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: DRAM_SDODT0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: DRAM_SDODT0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: DRAM_SDODT0.  0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field Select one of next values for pad: DRAM_SDODT0.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field Select one of next values for pad: DRAM_SDODT0.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ODT0 field descriptions (continued)**

Field	Description
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.204 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ODT1)

Address: 20E\_0000h base + 340h offset = 20E\_0340h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													DDR_SEL	DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	0			ODT		0		DSE			0
W																
Reset	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ODT1 field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ODT1 field descriptions (continued)**

Field	Description
	Select one of next values for pad: DRAM_SDODT1. 0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field Select one of next values for pad: DRAM_SDODT1. 0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: DRAM_SDODT1. 00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: DRAM_SDODT1. 0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: DRAM_SDODT1. 0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field Select one of next values for pad: DRAM_SDODT1. 000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field Select one of next values for pad: DRAM_SDODT1. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_ODT1 field descriptions (continued)**

Field	Description
010	<b>120_OHM</b> — 120 Ohm
011	<b>80_OHM</b> — 80 Ohm
100	<b>60_OHM</b> — 60 Ohm
101	<b>48_OHM</b> — 48 Ohm
110	<b>40_OHM</b> — 40 Ohm
111	<b>34_OHM</b> — 34 Ohm
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.205 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDQS0\_P)

Address: 20E\_0000h base + 344h offset = 20E\_0344h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0	DSE				0		
W																
Reset	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDQS0\_P field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRMODE_CTL  Note: The value of this field does not reflect the value of the Group Control Register.
16 HYS	Hysteresis Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRHYS  Note: The value of this field does not reflect the value of the Group Control Register.
15–14 PUS	Pull Up / Down Config. Field  Select one of next values for pad: DRAM_SDQS0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field  Select one of next values for pad: DRAM_SDQS0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field  Select one of next values for pad: DRAM_SDQS0.  0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Read Only Field  The value of this field is fixed and cannot be changed.  000 <b>DISABLED</b> — Disabled
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  Select one of next values for pad: DRAM_SDQS0.  000 <b>HIZ</b> — HI-Z

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDQS0\_P field descriptions (continued)**

Field	Description
001	<b>240_OHM</b> — 240 Ohm
010	<b>120_OHM</b> — 120 Ohm
011	<b>80_OHM</b> — 80 Ohm
100	<b>60_OHM</b> — 60 Ohm
101	<b>48_OHM</b> — 48 Ohm
110	<b>40_OHM</b> — 40 Ohm
111	<b>34_OHM</b> — 34 Ohm
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.206 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDQS1\_P)

Address: 20E\_0000h base + 348h offset = 20E\_0348h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0	DSE				0		
W																
Reset	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDQS1\_P field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRMODE_CTL  Note: The value of this field does not reflect the value of the Group Control Register.
16 HYS	Hysteresis Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRHYS  Note: The value of this field does not reflect the value of the Group Control Register.
15–14 PUS	Pull Up / Down Config. Field  Select one of next values for pad: DRAM_SDQS1.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field  Select one of next values for pad: DRAM_SDQS1.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field  Select one of next values for pad: DRAM_SDQS1.  0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Read Only Field  The value of this field is fixed and cannot be changed.  000 <b>DISABLED</b> — Disabled
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  Select one of next values for pad: DRAM_SDQS1.  000 <b>HIZ</b> — HI-Z

*Table continues on the next page...*



**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDQS1\_P field descriptions (continued)**

Field	Description
001	<b>240_OHM</b> — 240 Ohm
010	<b>120_OHM</b> — 120 Ohm
011	<b>80_OHM</b> — 80 Ohm
100	<b>60_OHM</b> — 60 Ohm
101	<b>48_OHM</b> — 48 Ohm
110	<b>40_OHM</b> — 40 Ohm
111	<b>34_OHM</b> — 34 Ohm
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.207 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDQS2\_P)

Address: 20E\_0000h base + 34Ch offset = 20E\_034Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													DDR_SEL	DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0	DSE				0		
W																
Reset	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDQS2\_P field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRMODE_CTL  Note: The value of this field does not reflect the value of the Group Control Register.
16 HYS	Hysteresis Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRHYS  Note: The value of this field does not reflect the value of the Group Control Register.
15–14 PUS	Pull Up / Down Config. Field  Select one of next values for pad: DRAM_SDQS2.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field  Select one of next values for pad: DRAM_SDQS2.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field  Select one of next values for pad: DRAM_SDQS2.  0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Read Only Field  The value of this field is fixed and cannot be changed.  000 <b>DISABLED</b> — Disabled
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  Select one of next values for pad: DRAM_SDQS2.  000 <b>HIZ</b> — HI-Z

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDQS2\_P field descriptions (continued)**

Field	Description
001	<b>240_OHM</b> — 240 Ohm
010	<b>120_OHM</b> — 120 Ohm
011	<b>80_OHM</b> — 80 Ohm
100	<b>60_OHM</b> — 60 Ohm
101	<b>48_OHM</b> — 48 Ohm
110	<b>40_OHM</b> — 40 Ohm
111	<b>34_OHM</b> — 34 Ohm
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.208 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDQS3\_P)

Address: 20E\_0000h base + 350h offset = 20E\_0350h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0	DSE				0		
W																
Reset	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDQS3\_P field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRMODE_CTL  Note: The value of this field does not reflect the value of the Group Control Register.
16 HYS	Hysteresis Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRHYS  Note: The value of this field does not reflect the value of the Group Control Register.
15–14 PUS	Pull Up / Down Config. Field  Select one of next values for pad: DRAM_SDQS3.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field  Select one of next values for pad: DRAM_SDQS3.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field  Select one of next values for pad: DRAM_SDQS3.  0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Read Only Field  The value of this field is fixed and cannot be changed.  000 <b>DISABLED</b> — Disabled
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field  Select one of next values for pad: DRAM_SDQS3.  000 <b>HIZ</b> — HI-Z

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDQS3\_P field descriptions (continued)**

Field	Description
001	<b>240_OHM</b> — 240 Ohm
010	<b>120_OHM</b> — 120 Ohm
011	<b>80_OHM</b> — 80 Ohm
100	<b>60_OHM</b> — 60 Ohm
101	<b>48_OHM</b> — 48 Ohm
110	<b>40_OHM</b> — 40 Ohm
111	<b>34_OHM</b> — 34 Ohm
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.209 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDWE\_B)

Address: 20E\_0000h base + 354h offset = 20E\_0354h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0	DSE			0			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDWE\_B field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDR_TYPE  Note: The value of this field does not reflect the value of the Group Control Register.
17 DDR_INPUT	DDR / CMOS Input Mode Field  Select one of next values for pad: DRAM_SDWE.  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: DRAM_SDWE.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Read Only Field  The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPK  Note: The value of this field does not reflect the value of the Group Control Register.
12 PKE	Pull / Keep Enable Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_DDRPKE  Note: The value of this field does not reflect the value of the Group Control Register.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field  Select one of next values for pad: DRAM_SDWE.  000 <b>DISABLED</b> — Disabled 001 <b>120_OHM</b> — 120 Ohm ODT 010 <b>60_OHM</b> — 60 Ohm ODT 011 <b>40_OHM</b> — 40 Ohm ODT 100 <b>30_OHM</b> — 30 Ohm ODT 101 <b>24_OHM</b> — 24 Ohm ODT 110 <b>20_OHM</b> — 20 Ohm ODT 111 <b>17_OHM</b> — 17 Ohm ODT
7–6 Reserved	This read-only field is reserved and always has the value 0.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_DRAM\_SDWE\_B field descriptions (continued)**

Field	Description
5–3 DSE	Drive Strength Field  This property can be configured using Group Control Register: IOMUXC_SW_PAD_CTL_GRP_CTLDS Note: The value of this field does not reflect the value of the Group Control Register.
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.210 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSP11\_MISO)

Address: 20E\_0000h base + 358h offset = 20E\_0358h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE		0		SRE	
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSP11\_MISO field descriptions**

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field  Select one of next values for pad: ECSP11_MISO.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field  Select one of next values for pad: ECSP11_MISO.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field  Select one of next values for pad: ECSP11_MISO.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up

*Table continues on the next page...*

## IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSP11\_MISO field descriptions (continued)

Field	Description
	10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: ECSP11_MISO. 0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: ECSP11_MISO. 0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: ECSP11_MISO. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: ECSP11_MISO. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.211 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSP11\_MOSI)

Address: 20E\_0000h base + 35Ch offset = 20E\_035Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSP11\_MOSI field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: ECSP11_MOSI.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: ECSP11_MOSI.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: ECSP11_MOSI.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: ECSP11_MOSI.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: ECSP11_MOSI.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSP11\_MOSI field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: ECSP11_MOSI. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: ECSP11_MOSI. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.212 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSP11\_SCLK)

Address: 20E\_0000h base + 360h offset = 20E\_0360h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSP11\_SCLK field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: ECSP11_SCLK.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: ECSP11_SCLK.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: ECSP11_SCLK.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: ECSP11_SCLK.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: ECSP11_SCLK.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSP11\_SCLK field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: ECSP11_SCLK.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: ECSP11_SCLK.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.213 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSP1\_SS0)

Address: 20E\_0000h base + 364h offset = 20E\_0364h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSP1\_SS0 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: ECSP1_SS0.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: ECSP1_SS0.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: ECSP1_SS0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: ECSP1_SS0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: ECSP1_SS0.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSP11\_SS0 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: ECSP11_SS0.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: ECSP11_SS0.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.214 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSPi2\_MISO)

Address: 20E\_0000h base + 368h offset = 20E\_0368h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSPi2\_MISO field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: ECSPi2_MISO.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: ECSPi2_MISO.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: ECSPi2_MISO.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: ECSPi2_MISO.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: ECSPi2_MISO.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSPi2\_MISO field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: ECSPi2_MISO.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: ECSPi2_MISO.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.215 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSPi2\_MOSI)

Address: 20E\_0000h base + 36Ch offset = 20E\_036Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	ODE	0		SPEED		DSE		0		SRE	
W																
Reset	0	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSPi2\_MOSI field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: ECSPi2_MOSI.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: ECSPi2_MOSI.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: ECSPi2_MOSI.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: ECSPi2_MOSI.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: ECSPi2_MOSI.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSPi2\_MOSI field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: ECSPi2_MOSI.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: ECSPi2_MOSI.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.216 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSPi2\_SCLK)

Address: 20E\_0000h base + 370h offset = 20E\_0370h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSPi2\_SCLK field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: ECSPi2_SCLK.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: ECSPi2_SCLK.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: ECSPi2_SCLK.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: ECSPi2_SCLK.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: ECSPi2_SCLK.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSPi2\_SCLK field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: ECSPi2_SCLK.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: ECSPi2_SCLK.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.217 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSPi2\_SS0)

Address: 20E\_0000h base + 374h offset = 20E\_0374h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSPi2\_SS0 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: ECSPi2_SS0.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: ECSPi2_SS0.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: ECSPi2_SS0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: ECSPi2_SS0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: ECSPi2_SS0.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_ECSPi2\_SS0 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: ECSPi2_SS0.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: ECSPi2_SS0.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.218 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_BDR0)

Address: 20E\_0000h base + 378h offset = 20E\_0378h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R								0		SPEED		DSE			0	
W	PUS		PUE	PKE	ODE											SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_BDR0 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_BDR0.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_BDR0.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_BDR0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_BDR0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_BDR0.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_BDR0 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_BDR0.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_BDR0.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.219 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_BDR1)

Address: 20E\_0000h base + 37Ch offset = 20E\_037Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_BDR1 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_BDR1.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_BDR1.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_BDR1.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_BDR1.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_BDR1.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_BDR1 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_BDR1. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_BDR1. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.220 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA00)

Address: 20E\_0000h base + 380h offset = 20E\_0380h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA00 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_D0.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_D0.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_D0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_D0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_D0.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA00 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_D0.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_D0.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.221 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA01)

Address: 20E\_0000h base + 384h offset = 20E\_0384h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA01 field descriptions**

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_D1.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_D1.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_D1.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_D1.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_D1.

*Table continues on the next page...*

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA01 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_D1.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_D1.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.222 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA10)

Address: 20E\_0000h base + 388h offset = 20E\_0388h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA10 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_D10.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_D10.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_D10.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_D10.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_D10.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA10 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_D10.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_D10.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.223 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA11)

Address: 20E\_0000h base + 38Ch offset = 20E\_038Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA11 field descriptions**

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_D11.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_D11.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_D11.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_D11.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_D11.

*Table continues on the next page...*

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA11 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_D11.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_D11.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.224 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA12)

Address: 20E\_0000h base + 390h offset = 20E\_0390h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA12 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_D12.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_D12.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_D12.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_D12.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_D12.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA12 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_D12.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_D12.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.225 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA13)

Address: 20E\_0000h base + 394h offset = 20E\_0394h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA13 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_D13.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_D13.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_D13.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_D13.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_D13.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA13 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_D13.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_D13.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.226 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA14)

Address: 20E\_0000h base + 398h offset = 20E\_0398h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA14 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_D14.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_D14.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_D14.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_D14.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_D14.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA14 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_D14.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_D14.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.227 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA15)

Address: 20E\_0000h base + 39Ch offset = 20E\_039Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS	PUE	PKE	ODE	0				SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA15 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_D15.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_D15.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_D15.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_D15.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_D15.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA15 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_D15.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_D15.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.228 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA02)

Address: 20E\_0000h base + 3A0h offset = 20E\_03A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA02 field descriptions**

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_D2. 0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_D2. 0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_D2. 00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_D2. 0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_D2.

*Table continues on the next page...*

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA02 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_D2. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_D2. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.229 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA03)

Address: 20E\_0000h base + 3A4h offset = 20E\_03A4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA03 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_D3.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_D3.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_D3.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_D3.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_D3.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA03 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_D3.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_D3.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.230 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA04)

Address: 20E\_0000h base + 3A8h offset = 20E\_03A8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA04 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_D4.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_D4.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_D4.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_D4.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_D4.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA04 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_D4.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_D4.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.231 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA05)

Address: 20E\_0000h base + 3ACh offset = 20E\_03ACh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA05 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_D5.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_D5.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_D5.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_D5.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_D5.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA05 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_D5.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_D5.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.232 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA06)

Address: 20E\_0000h base + 3B0h offset = 20E\_03B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA06 field descriptions**

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_D6.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_D6.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_D6.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_D6.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_D6.

*Table continues on the next page...*

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA06 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_D6.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_D6.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.233 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA07)

Address: 20E\_0000h base + 3B4h offset = 20E\_03B4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA07 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_D7.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_D7.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_D7.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_D7.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_D7.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA07 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_D7.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_D7.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.234 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA08)

Address: 20E\_0000h base + 3B8h offset = 20E\_03B8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA08 field descriptions**

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_D8.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_D8.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_D8.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_D8.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_D8.

*Table continues on the next page...*

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA08 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_D8.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_D8.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.235 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA09)

Address: 20E\_0000h base + 3BCh offset = 20E\_03BCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA09 field descriptions**

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_D9.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_D9.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_D9.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_D9.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_D9.

*Table continues on the next page...*

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_DATA09 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_D9.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_D9.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.236 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_GDCLK)

Address: 20E\_0000h base + 3C0h offset = 20E\_03C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_GDCLK field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_GDCLK.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_GDCLK.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_GDCLK.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_GDCLK.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_GDCLK.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_GDCLK field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_GDCLK.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_GDCLK.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.237 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_GDOE)

Address: 20E\_0000h base + 3C4h offset = 20E\_03C4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		
W																SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_GDOE field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_GDOE.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_GDOE.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_GDOE.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_GDOE.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_GDOE.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_GDOE field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_GDOE.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_GDOE.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.238 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_GDRL)

Address: 20E\_0000h base + 3C8h offset = 20E\_03C8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_GDRL field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_GDRL.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_GDRL.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_GDRL.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_GDRL.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_GDRL.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_GDRL field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_GDRL.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_GDRL.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.239 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_GDSP)

Address: 20E\_0000h base + 3CCh offset = 20E\_03CCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_GDSP field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_GDSP.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_GDSP.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_GDSP.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_GDSP.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_GDSP.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_GDSP field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_GDSP.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_GDSP.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.240 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_COM)

Address: 20E\_0000h base + 3D0h offset = 20E\_03D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE	0					HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS	PUE	PKE	ODE	0				SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_COM field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_PWRCOM.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_PWRCOM.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_PWRCOM.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_PWRCOM.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_PWRCOM.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_COM field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_PWRCOM.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_PWRCOM.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.241 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_CTRL0)

Address: 20E\_0000h base + 3D4h offset = 20E\_03D4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	ODE	0		SPEED		DSE		0		SRE	
W																
Reset	0	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_CTRL0 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_PWRCTRL0.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_PWRCTRL0.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_PWRCTRL0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_PWRCTRL0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_PWRCTRL0.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_CTRL0 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_PWRCTRL0.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_PWRCTRL0.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.242 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_CTRL1)

Address: 20E\_0000h base + 3D8h offset = 20E\_03D8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_CTRL1 field descriptions**

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_PWRCTRL1.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_PWRCTRL1.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_PWRCTRL1.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_PWRCTRL1.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_PWRCTRL1.

*Table continues on the next page...*

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_CTRL1 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_PWRCTRL1.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_PWRCTRL1.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.243 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_CTRL2)

Address: 20E\_0000h base + 3DCh offset = 20E\_03DCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_CTRL2 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_PWRCTRL2.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_PWRCTRL2.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_PWRCTRL2.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_PWRCTRL2.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_PWRCTRL2.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_CTRL2 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_PWRCTRL2.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_PWRCTRL2.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.244 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_CTRL3)

Address: 20E\_0000h base + 3E0h offset = 20E\_03E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	ODE	0		SPEED		DSE		0		SRE	
W																
Reset	0	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_CTRL3 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_PWRCTRL3.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_PWRCTRL3.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_PWRCTRL3.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_PWRCTRL3.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_PWRCTRL3.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_CTRL3 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_PWRCTRL3.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_PWRCTRL3.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.245 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_IRQ)

Address: 20E\_0000h base + 3E4h offset = 20E\_03E4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_IRQ field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_PWRINT.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_PWRINT.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_PWRINT.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_PWRINT.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_PWRINT.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_IRQ field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_PWRINT.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_PWRINT.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.246 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_STAT)

Address: 20E\_0000h base + 3E8h offset = 20E\_03E8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_STAT field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_PWRSTAT.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_PWRSTAT.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_PWRSTAT.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_PWRSTAT.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_PWRSTAT.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_STAT field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_PWRSTAT.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_PWRSTAT.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.247 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_WAKE)

Address: 20E\_0000h base + 3ECh offset = 20E\_03ECh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
R	0								LVE		0						HYS	
W																		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1		

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_WAKE field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_PWRWAKEUP.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_PWRWAKEUP.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_PWRWAKEUP.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_PWRWAKEUP.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_PWRWAKEUP.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_PWR\_WAKE field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_PWRWAKEUP.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_PWRWAKEUP.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.248 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDCE0)

Address: 20E\_0000h base + 3F0h offset = 20E\_03F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDCE0 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_SDCE0.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_SDCE0.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_SDCE0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_SDCE0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_SDCE0.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDCE0 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_SDCE0.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_SDCE0.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.249 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDCE1)

Address: 20E\_0000h base + 3F4h offset = 20E\_03F4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDCE1 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_SDCE1.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_SDCE1.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_SDCE1.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_SDCE1.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_SDCE1.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDCE1 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_SDCE1.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_SDCE1.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.250 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDCE2)

Address: 20E\_0000h base + 3F8h offset = 20E\_03F8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDCE2 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_SDCE2.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_SDCE2.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_SDCE2.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_SDCE2.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_SDCE2.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDCE2 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_SDCE2.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_SDCE2.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.251 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDCE3)

Address: 20E\_0000h base + 3FCh offset = 20E\_03FCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	ODE	0		SPEED		DSE		0		SRE	
W																
Reset	0	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDCE3 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_SDCE3.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_SDCE3.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_SDCE3.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_SDCE3.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_SDCE3.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDCE3 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_SDCE3.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_SDCE3.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.252 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDCLK)

Address: 20E\_0000h base + 400h offset = 20E\_0400h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDCLK field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_SDCLK.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_SDCLK.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_SDCLK.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_SDCLK.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_SDCLK.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDCLK field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_SDCLK.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_SDCLK.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.253 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDLE)

Address: 20E\_0000h base + 404h offset = 20E\_0404h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDLE field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_SDLE.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_SDLE.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_SDLE.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_SDLE.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_SDLE.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDLE field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_SDLE. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_SDLE. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.254 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDOE)

Address: 20E\_0000h base + 408h offset = 20E\_0408h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		
W																SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDOE field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_SDOE.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_SDOE.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_SDOE.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_SDOE.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_SDOE.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDOE field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_SDOE. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_SDOE. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.255 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDSHR)

Address: 20E\_0000h base + 40Ch offset = 20E\_040Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDSHR field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_SDSHR.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_SDSHR.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_SDSHR.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_SDSHR.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_SDSHR.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_SDSHR field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_SDSHR. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_SDSHR. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.256 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_VCOM0)

Address: 20E\_0000h base + 410h offset = 20E\_0410h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	ODE	0		SPEED		DSE		0		SRE	
W																
Reset	0	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_VCOM0 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_VCOM0.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_VCOM0.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_VCOM0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_VCOM0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_VCOM0.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_VCOM0 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_VCOM0. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_VCOM0. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.257 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_VCOM1)

Address: 20E\_0000h base + 414h offset = 20E\_0414h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_VCOM1 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: EPDC_VCOM1.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: EPDC_VCOM1.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: EPDC_VCOM1.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: EPDC_VCOM1.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: EPDC_VCOM1.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_EPDC\_VCOM1 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: EPDC_VCOM1. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: EPDC_VCOM1. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.258 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_CRS\_DV)

Address: 20E\_0000h base + 418h offset = 20E\_0418h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_CRS\_DV field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: FEC_CRS_DV.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: FEC_CRS_DV.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: FEC_CRS_DV.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: FEC_CRS_DV.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: FEC_CRS_DV.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_CRS\_DV field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: FEC_CRS_DV.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: FEC_CRS_DV.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.259 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_MDC)

Address: 20E\_0000h base + 41Ch offset = 20E\_041Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE	0					
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	ODE	0		SPEED		DSE		0		SRE	
W																
Reset	0	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_MDC field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: FEC_MDC.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: FEC_MDC.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: FEC_MDC.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: FEC_MDC.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: FEC_MDC.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_MDC field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: FEC_MDC. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: FEC_MDC. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.260 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_MDIO)

Address: 20E\_0000h base + 420h offset = 20E\_0420h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_MDIO field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: FEC_MDIO.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: FEC_MDIO.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: FEC_MDIO.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: FEC_MDIO.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: FEC_MDIO.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_MDIO field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: FEC_MDIO. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: FEC_MDIO. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.261 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_REF\_CLK)

Address: 20E\_0000h base + 424h offset = 20E\_0424h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_REF\_CLK field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: FEC_REF_CLK.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: FEC_REF_CLK.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: FEC_REF_CLK.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: FEC_REF_CLK.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: FEC_REF_CLK.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_REF\_CLK field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: FEC_REF_CLK. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: FEC_REF_CLK. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.262 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_RX\_ER)

Address: 20E\_0000h base + 428h offset = 20E\_0428h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_RX\_ER field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: FEC_RX_ER.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: FEC_RX_ER.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: FEC_RX_ER.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: FEC_RX_ER.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: FEC_RX_ER.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_RX\_ER field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: FEC_RX_ER. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: FEC_RX_ER. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.263 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_RX\_DATA0)

Address: 20E\_0000h base + 42Ch offset = 20E\_042Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_RX\_DATA0 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: FEC_RXD0.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: FEC_RXD0.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: FEC_RXD0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: FEC_RXD0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: FEC_RXD0.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_RX\_DATA0 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: FEC_RXD0. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: FEC_RXD0. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.264 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_RX\_DATA1)

Address: 20E\_0000h base + 430h offset = 20E\_0430h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_RX\_DATA1 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: FEC_RXD1.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: FEC_RXD1.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: FEC_RXD1.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: FEC_RXD1.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: FEC_RXD1.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_RX\_DATA1 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: FEC_RXD1.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: FEC_RXD1.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.265 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_TX\_CLK)

Address: 20E\_0000h base + 434h offset = 20E\_0434h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_TX\_CLK field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: FEC_TX_CLK.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: FEC_TX_CLK.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: FEC_TX_CLK.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: FEC_TX_CLK.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: FEC_TX_CLK.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_TX\_CLK field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: FEC_TX_CLK. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: FEC_TX_CLK. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.266 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_TX\_EN)

Address: 20E\_0000h base + 438h offset = 20E\_0438h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_TX\_EN field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: FEC_TX_EN.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: FEC_TX_EN.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: FEC_TX_EN.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: FEC_TX_EN.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: FEC_TX_EN.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_TX\_EN field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: FEC_TX_EN. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: FEC_TX_EN. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.267 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_TX\_DATA0)

Address: 20E\_0000h base + 43Ch offset = 20E\_043Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_TX\_DATA0 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: FEC_TXD0.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: FEC_TXD0.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: FEC_TXD0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: FEC_TXD0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: FEC_TXD0.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_TX\_DATA0 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: FEC_TXD0. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: FEC_TXD0. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.268 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_TX\_DATA1)

Address: 20E\_0000h base + 440h offset = 20E\_0440h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_TX\_DATA1 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: FEC_TXD1.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: FEC_TXD1.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: FEC_TXD1.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: FEC_TXD1.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: FEC_TXD1.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_FEC\_TX\_DATA1 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: FEC_TXD1. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: FEC_TXD1. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.269 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_USB\_H\_DATA)

Address: 20E\_0000h base + 444h offset = 20E\_0444h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0	DSE				0		
W																
Reset	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_USB\_H\_DATA field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field Select one of next values for pad: HSIC_DAT.  00 <b>RESERVED0</b> — Reserved 01 <b>RESERVED1</b> — Reserved 10 <b>1P2V_IO</b> — 1.2V I/O interfaces including USB HSIC. Provides calibrated drive strengths for signals ranging from 1.0V up to 1.3V. 11 <b>1P5V_IO</b> — 1.5V I/O interfaces. Provides calibrated drive strengths for signals ranging from 1.3V to 2.5V.
17 DDR_INPUT	DDR / CMOS Input Mode Field Select one of next values for pad: HSIC_DAT.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_USB\_H\_DATA field descriptions (continued)**

Field	Description
	0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field Select one of next values for pad: HSIC_DAT. 0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: HSIC_DAT. 00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: HSIC_DAT. 0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: HSIC_DAT. 0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field Read Only Field The value of this field is fixed and cannot be changed. 000 <b>DISABLED</b> — Disabled
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field Select one of next values for pad: HSIC_DAT. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_USB\_H\_DATA field descriptions (continued)**

Field	Description
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.270 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_USB\_H\_STROBE)

Address: 20E\_0000h base + 448h offset = 20E\_0448h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		DDR_INPUT	HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	0	ODT			0	DSE				0		
W																
Reset	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_USB\_H\_STROBE field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	DDR Select Field Select one of next values for pad: HSIC_STROBE.  00 <b>RESERVED0</b> — Reserved 01 <b>RESERVED1</b> — Reserved

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_USB\_H\_STROBE field descriptions (continued)**

Field	Description
	10 <b>1P2V_IO</b> — 1.2V I/O interfaces including USB HSIC. Provides calibrated drive strengths for signals ranging from 1.0V up to 1.3V. 11 <b>1P5V_IO</b> — 1.5V I/O interfaces. Provides calibrated drive strengths for signals ranging from 1.3V to 2.5V.
17 DDR_INPUT	DDR / CMOS Input Mode Field Select one of next values for pad: HSIC_STROBE. 0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16 HYS	Hysteresis Enable Field Select one of next values for pad: HSIC_STROBE. 0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: HSIC_STROBE. 00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: HSIC_STROBE. 0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: HSIC_STROBE. 0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT	On Die Termination Field Read Only Field The value of this field is fixed and cannot be changed. 000 <b>DISABLED</b> — Disabled
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	Drive Strength Field Select one of next values for pad: HSIC_STROBE. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm

*Table continues on the next page...*



**IOMUXC\_SW\_PAD\_CTL\_PAD\_USB\_H\_STROBE field descriptions (continued)**

Field	Description
010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm	
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.271 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_I2C1\_SCL)

Address: 20E\_0000h base + 44Ch offset = 20E\_044Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE		0		SRE	
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_I2C1\_SCL field descriptions**

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: I2C1_SCL.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: I2C1_SCL.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: I2C1_SCL.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_I2C1\_SCL field descriptions (continued)**

Field	Description
	00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: I2C1_SCL. 0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: I2C1_SCL. 0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: I2C1_SCL. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: I2C1_SCL. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_I2C1\_SCL field descriptions (continued)**

Field	Description
0	<b>SLOW</b> — Slow Slew Rate
1	<b>FAST</b> — Fast Slew Rate

### 30.4.272 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_I2C1\_SDA)

Address: 20E\_0000h base + 450h offset = 20E\_0450h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_I2C1\_SDA field descriptions**

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: I2C1_SDA.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: I2C1_SDA.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: I2C1_SDA.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: I2C1_SDA.

*Table continues on the next page...*

## IOMUXC\_SW\_PAD\_CTL\_PAD\_I2C1\_SDA field descriptions (continued)

Field	Description
	0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: I2C1_SDA. 0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: I2C1_SDA. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: I2C1_SDA. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.273 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_I2C2\_SCL)

Address: 20E\_0000h base + 454h offset = 20E\_0454h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_I2C2\_SCL field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: I2C2_SCL.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: I2C2_SCL.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: I2C2_SCL.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: I2C2_SCL.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: I2C2_SCL.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_I2C2\_SCL field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: I2C2_SCL.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: I2C2_SCL.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.274 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_I2C2\_SDA)

Address: 20E\_0000h base + 458h offset = 20E\_0458h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_I2C2\_SDA field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: I2C2_SDA.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: I2C2_SDA.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: I2C2_SDA.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: I2C2_SDA.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: I2C2_SDA.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_I2C2\_SDA field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: I2C2_SDA.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: I2C2_SDA.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.275 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_MOD)

Address: 20E\_0000h base + 45Ch offset = 20E\_045Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	0									LVE		0					HYS
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE		0		SRE	
W																
Reset	1	0	1	1	0	0	0	0	0	1	1	0	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_MOD field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Read Only Field The value of this field is fixed and cannot be changed.  0 <b>DISABLED</b> — High Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: JTAG_MOD.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: JTAG_MOD.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Read Only Field The value of this field is fixed and cannot be changed.  1 <b>PULL</b> — Pull Enabled

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_MOD field descriptions (continued)**

Field	Description
12 PKE	<p>Pull / Keep Enable Field</p> <p>Select one of next values for pad: JTAG_MOD.</p> <p>0 <b>DISABLED</b> — Pull/Keeper Disabled</p> <p>1 <b>ENABLED</b> — Pull/Keeper Enabled</p>
11 ODE	<p>Open Drain Enable Field</p> <p>Enables open drain of the pin.</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>0 <b>DISABLED</b> — Output is CMOS.</p>
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	<p>Speed Field</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>01 <b>50MHZ</b> — Low (50 MHz)</p>
5–3 DSE	<p>Drive Strength Field</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>100 <b>60_OHM</b> — 60 Ohm</p>
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	<p>Slew Rate Field</p> <p>Slew rate control.</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>0 <b>SLOW</b> — Slow Slew Rate</p>

### 30.4.276 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_TCK)

Address: 20E\_0000h base + 460h offset = 20E\_0460h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	0									LVE		0					HYS
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE		0		SRE	
W																
Reset	0	1	1	1	0	0	0	0	0	1	1	0	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_TCK field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Read Only Field The value of this field is fixed and cannot be changed.  0 <b>DISABLED</b> — High Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: JTAG_TCK.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: JTAG_TCK.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Read Only Field The value of this field is fixed and cannot be changed.  1 <b>PULL</b> — Pull Enabled

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_TCK field descriptions (continued)**

Field	Description
12 PKE	<p>Pull / Keep Enable Field</p> <p>Select one of next values for pad: JTAG_TCK.</p> <p>0 <b>DISABLED</b> — Pull/Keeper Disabled</p> <p>1 <b>ENABLED</b> — Pull/Keeper Enabled</p>
11 ODE	<p>Open Drain Enable Field</p> <p>Enables open drain of the pin.</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>0 <b>DISABLED</b> — Output is CMOS.</p>
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	<p>Speed Field</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>01 <b>50MHZ</b> — Low (50 MHz)</p>
5–3 DSE	<p>Drive Strength Field</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>100 <b>60_OHM</b> — 60 Ohm</p>
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	<p>Slew Rate Field</p> <p>Slew rate control.</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>0 <b>SLOW</b> — Slow Slew Rate</p>

### 30.4.277 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_TDI)

Address: 20E\_0000h base + 464h offset = 20E\_0464h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	0									LVE		0					HYS
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE		0		SRE	
W																
Reset	0	1	1	1	0	0	0	0	0	1	1	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_TDI field descriptions**

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Read Only Field The value of this field is fixed and cannot be changed.  0 <b>DISABLED</b> — High Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: JTAG_TDI.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: JTAG_TDI.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Read Only Field The value of this field is fixed and cannot be changed.  1 <b>PULL</b> — Pull Enabled

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_TDI field descriptions (continued)**

Field	Description
12 PKE	<p>Pull / Keep Enable Field</p> <p>Select one of next values for pad: JTAG_TDI.</p> <p>0 <b>DISABLED</b> — Pull/Keeper Disabled</p> <p>1 <b>ENABLED</b> — Pull/Keeper Enabled</p>
11 ODE	<p>Open Drain Enable Field</p> <p>Enables open drain of the pin.</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>0 <b>DISABLED</b> — Output is CMOS.</p>
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	<p>Speed Field</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>01 <b>50MHZ</b> — Low (50 MHz)</p>
5–3 DSE	<p>Drive Strength Field</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>100 <b>60_OHM</b> — 60 Ohm</p>
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	<p>Slew Rate Field</p> <p>Slew rate control.</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>0 <b>SLOW</b> — Slow Slew Rate</p>

### 30.4.278 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_TDO)

Address: 20E\_0000h base + 468h offset = 20E\_0468h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	0									LVE	0						HYS
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE		0		SRE	
W																
Reset	1	0	0	1	0	0	0	0	1	0	1	1	0	0	0	1

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_TDO field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Read Only Field The value of this field is fixed and cannot be changed.  0 <b>DISABLED</b> — High Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Read Only Field The value of this field is fixed and cannot be changed.  0 <b>DISABLED</b> — CMOS input
15–14 PUS	Pull Up / Down Config. Field Read Only Field The value of this field is fixed and cannot be changed.  10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up
13 PUE	Pull / Keep Select Field Read Only Field The value of this field is fixed and cannot be changed.  0 <b>KEEP</b> — Keeper Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: JTAG_TDO.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_TDO field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. Read Only Field The value of this field is fixed and cannot be changed. 0 <b>DISABLED</b> — Output is CMOS.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Read Only Field The value of this field is fixed and cannot be changed. 10 <b>100MHZ</b> — Medium (100 MHz)
5–3 DSE	Drive Strength Field Read Only Field The value of this field is fixed and cannot be changed. 110 <b>40_OHM</b> — 40 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. Read Only Field The value of this field is fixed and cannot be changed. 1 <b>FAST</b> — Fast Slew Rate



### 30.4.279 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_TMS)

Address: 20E\_0000h base + 46Ch offset = 20E\_046Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	0									LVE		0					HYS
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE		0		SRE	
W																
Reset	0	1	1	1	0	0	0	0	0	1	1	0	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_TMS field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Read Only Field The value of this field is fixed and cannot be changed.  0 <b>DISABLED</b> — High Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: JTAG_TMS.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: JTAG_TMS.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Read Only Field The value of this field is fixed and cannot be changed.  1 <b>PULL</b> — Pull Enabled

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_TMS field descriptions (continued)**

Field	Description
12 PKE	<p>Pull / Keep Enable Field</p> <p>Select one of next values for pad: JTAG_TMS.</p> <p>0 <b>DISABLED</b> — Pull/Keeper Disabled</p> <p>1 <b>ENABLED</b> — Pull/Keeper Enabled</p>
11 ODE	<p>Open Drain Enable Field</p> <p>Enables open drain of the pin.</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>0 <b>DISABLED</b> — Output is CMOS.</p>
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	<p>Speed Field</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>01 <b>50MHZ</b> — Low (50 MHz)</p>
5–3 DSE	<p>Drive Strength Field</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>100 <b>60_OHM</b> — 60 Ohm</p>
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	<p>Slew Rate Field</p> <p>Slew rate control.</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>0 <b>SLOW</b> — Slow Slew Rate</p>

### 30.4.280 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_TRSTB)

Address: 20E\_0000h base + 470h offset = 20E\_0470h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	0									LVE		0					HYS
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE		0		SRE	
W																
Reset	0	1	1	1	0	0	0	0	0	1	1	0	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_TRSTB field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Read Only Field The value of this field is fixed and cannot be changed.  0 <b>DISABLED</b> — High Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: JTAG_TRSTB.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: JTAG_TRSTB.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Read Only Field The value of this field is fixed and cannot be changed.  1 <b>PULL</b> — Pull Enabled

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_JTAG\_TRSTB field descriptions (continued)**

Field	Description
12 PKE	<p>Pull / Keep Enable Field</p> <p>Select one of next values for pad: JTAG_TRSTB.</p> <p>0 <b>DISABLED</b> — Pull/Keeper Disabled</p> <p>1 <b>ENABLED</b> — Pull/Keeper Enabled</p>
11 ODE	<p>Open Drain Enable Field</p> <p>Enables open drain of the pin.</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>0 <b>DISABLED</b> — Output is CMOS.</p>
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	<p>Speed Field</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>01 <b>50MHZ</b> — Low (50 MHz)</p>
5–3 DSE	<p>Drive Strength Field</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>100 <b>60_OHM</b> — 60 Ohm</p>
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	<p>Slew Rate Field</p> <p>Slew rate control.</p> <p>Read Only Field</p> <p>The value of this field is fixed and cannot be changed.</p> <p>0 <b>SLOW</b> — Slow Slew Rate</p>

### 30.4.281 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL0)

Address: 20E\_0000h base + 474h offset = 20E\_0474h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE	0					HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS	PUE	PKE	ODE	0				SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL0 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: KEY_COL0.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: KEY_COL0.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: KEY_COL0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: KEY_COL0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: KEY_COL0.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL0 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: KEY_COL0. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: KEY_COL0. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.282 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL1)

Address: 20E\_0000h base + 478h offset = 20E\_0478h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL1 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: KEY_COL1.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: KEY_COL1.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: KEY_COL1.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: KEY_COL1.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: KEY_COL1.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL1 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: KEY_COL1.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: KEY_COL1.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.283 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL2)

Address: 20E\_0000h base + 47Ch offset = 20E\_047Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL2 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: KEY_COL2.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: KEY_COL2.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: KEY_COL2.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: KEY_COL2.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: KEY_COL2.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL2 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: KEY_COL2.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: KEY_COL2.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.284 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL3)

Address: 20E\_0000h base + 480h offset = 20E\_0480h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL3 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: KEY_COL3.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: KEY_COL3.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: KEY_COL3.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: KEY_COL3.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: KEY_COL3.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL3 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: KEY_COL3. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: KEY_COL3. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.285 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL4)

Address: 20E\_0000h base + 484h offset = 20E\_0484h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE	0					HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE		0			SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL4 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: KEY_COL4.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: KEY_COL4.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: KEY_COL4.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: KEY_COL4.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: KEY_COL4.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL4 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: KEY_COL4. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: KEY_COL4. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.286 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL5)

Address: 20E\_0000h base + 488h offset = 20E\_0488h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL5 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: KEY_COL5.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: KEY_COL5.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: KEY_COL5.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: KEY_COL5.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: KEY_COL5.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL5 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: KEY_COL5. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: KEY_COL5. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.287 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL6)

Address: 20E\_0000h base + 48Ch offset = 20E\_048Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	0								LVE		0					HYS	
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL6 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: KEY_COL6.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: KEY_COL6.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: KEY_COL6.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: KEY_COL6.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: KEY_COL6.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL6 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: KEY_COL6. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: KEY_COL6. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.288 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL7)

Address: 20E\_0000h base + 490h offset = 20E\_0490h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL7 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: KEY_COL7.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: KEY_COL7.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: KEY_COL7.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: KEY_COL7.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: KEY_COL7.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_COL7 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: KEY_COL7. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: KEY_COL7. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.289 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW0)

Address: 20E\_0000h base + 494h offset = 20E\_0494h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW0 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: KEY_ROW0.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: KEY_ROW0.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: KEY_ROW0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: KEY_ROW0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: KEY_ROW0.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW0 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: KEY_ROW0.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: KEY_ROW0.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.290 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW1)

Address: 20E\_0000h base + 498h offset = 20E\_0498h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW1 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: KEY_ROW1.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: KEY_ROW1.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: KEY_ROW1.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: KEY_ROW1.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: KEY_ROW1.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW1 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: KEY_ROW1. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: KEY_ROW1. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.291 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW2)

Address: 20E\_0000h base + 49Ch offset = 20E\_049Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW2 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: KEY_ROW2.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: KEY_ROW2.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: KEY_ROW2.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: KEY_ROW2.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: KEY_ROW2.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW2 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: KEY_ROW2. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: KEY_ROW2. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.292 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW3)

Address: 20E\_0000h base + 4A0h offset = 20E\_04A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW3 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: KEY_ROW3.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: KEY_ROW3.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: KEY_ROW3.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: KEY_ROW3.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: KEY_ROW3.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW3 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: KEY_ROW3. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: KEY_ROW3. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.293 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW4)

Address: 20E\_0000h base + 4A4h offset = 20E\_04A4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW4 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: KEY_ROW4.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: KEY_ROW4.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: KEY_ROW4.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: KEY_ROW4.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: KEY_ROW4.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW4 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: KEY_ROW4. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: KEY_ROW4. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.294 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW5)

Address: 20E\_0000h base + 4A8h offset = 20E\_04A8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW5 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: KEY_ROW5.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: KEY_ROW5.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: KEY_ROW5.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: KEY_ROW5.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: KEY_ROW5.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW5 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: KEY_ROW5.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: KEY_ROW5.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.295 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW6)

Address: 20E\_0000h base + 4ACh offset = 20E\_04ACh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW6 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: KEY_ROW6.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: KEY_ROW6.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: KEY_ROW6.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: KEY_ROW6.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: KEY_ROW6.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW6 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: KEY_ROW6.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: KEY_ROW6.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.296 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW7)

Address: 20E\_0000h base + 4B0h offset = 20E\_04B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW7 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: KEY_ROW7.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: KEY_ROW7.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: KEY_ROW7.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: KEY_ROW7.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: KEY_ROW7.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_KEY\_ROW7 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: KEY_ROW7. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: KEY_ROW7. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.297 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_CLK)

Address: 20E\_0000h base + 4B4h offset = 20E\_04B4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_CLK field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_CLK.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_CLK.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_CLK.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_CLK.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_CLK.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_CLK field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_CLK. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_CLK. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.298 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA0)

Address: 20E\_0000h base + 4B8h offset = 20E\_04B8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA00 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT0.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT0.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT0.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA00 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT0.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT0.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.299 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA01)

Address: 20E\_0000h base + 4BCh offset = 20E\_04BCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA01 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT1.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT1.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT1.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT1.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT1.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA01 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT1.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT1.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.300 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA10)

Address: 20E\_0000h base + 4C0h offset = 20E\_04C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA10 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT10.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT10.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT10.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT10.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT10.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA10 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT10.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT10.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.301 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA11)

Address: 20E\_0000h base + 4C4h offset = 20E\_04C4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA11 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT11.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT11.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT11.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT11.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT11.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA11 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT11. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT11. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.302 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA12)

Address: 20E\_0000h base + 4C8h offset = 20E\_04C8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	0								LVE		0						HYS
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0		SPEED		DSE			0		SRE	
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA12 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT12.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT12.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT12.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT12.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT12.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA12 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT12. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT12. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.303 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA13)

Address: 20E\_0000h base + 4CCh offset = 20E\_04CCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA13 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT13.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT13.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT13.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT13.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT13.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA13 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT13. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT13. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.304 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA14)

Address: 20E\_0000h base + 4D0h offset = 20E\_04D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA14 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT14.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT14.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT14.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT14.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT14.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA14 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT14. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT14. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.305 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA15)

Address: 20E\_0000h base + 4D4h offset = 20E\_04D4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA15 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT15.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT15.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT15.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT15.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT15.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA15 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT15.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT15.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.306 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA16)

Address: 20E\_0000h base + 4D8h offset = 20E\_04D8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	ODE	0		SPEED		DSE		0		SRE	
W																
Reset	0	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA16 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT16.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT16.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT16.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT16.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT16.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA16 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT16.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT16.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.307 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA17)

Address: 20E\_0000h base + 4DCh offset = 20E\_04DCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA17 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT17.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT17.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT17.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT17.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT17.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA17 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT17. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT17. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.308 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA18)

Address: 20E\_0000h base + 4E0h offset = 20E\_04E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA18 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT18.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT18.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT18.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT18.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT18.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA18 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT18.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT18.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.309 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA19)

Address: 20E\_0000h base + 4E4h offset = 20E\_04E4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA19 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT19.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT19.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT19.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT19.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT19.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA19 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT19.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT19.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.310 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA02)

Address: 20E\_0000h base + 4E8h offset = 20E\_04E8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA02 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT2.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT2.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT2.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT2.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT2.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA02 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT2. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT2. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.311 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA20)

Address: 20E\_0000h base + 4ECh offset = 20E\_04ECh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA20 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT20.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT20.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT20.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT20.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT20.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA20 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT20.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT20.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.312 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA21)

Address: 20E\_0000h base + 4F0h offset = 20E\_04F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA21 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT21.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT21.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT21.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT21.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT21.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA21 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT21.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT21.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.313 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA22)

Address: 20E\_0000h base + 4F4h offset = 20E\_04F4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		
W																SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA22 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT22.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT22.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT22.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT22.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT22.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA22 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT22.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT22.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.314 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA23)

Address: 20E\_0000h base + 4F8h offset = 20E\_04F8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA23 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT23.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT23.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT23.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT23.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT23.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA23 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT23. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT23. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.315 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA03)

Address: 20E\_0000h base + 4FCh offset = 20E\_04FCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA03 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT3.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT3.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT3.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT3.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT3.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA03 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT3.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT3.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.316 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA04)

Address: 20E\_0000h base + 500h offset = 20E\_0500h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	ODE	0		SPEED		DSE		0		SRE	
W																
Reset	0	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA04 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT4.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT4.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT4.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT4.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT4.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA04 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT4.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT4.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.317 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA05)

Address: 20E\_0000h base + 504h offset = 20E\_0504h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA05 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT5.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT5.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT5.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT5.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT5.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA05 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT5.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT5.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.318 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA06)

Address: 20E\_0000h base + 508h offset = 20E\_0508h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA06 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT6.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT6.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT6.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT6.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT6.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA06 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT6.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT6.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.319 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA07)

Address: 20E\_0000h base + 50Ch offset = 20E\_050Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA07 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT7.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT7.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT7.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT7.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT7.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA07 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT7.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT7.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.320 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA08)

Address: 20E\_0000h base + 510h offset = 20E\_0510h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA08 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT8.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT8.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT8.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT8.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT8.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA08 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT8. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT8. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.321 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA09)

Address: 20E\_0000h base + 514h offset = 20E\_0514h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA09 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_DAT9.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_DAT9.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_DAT9.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_DAT9.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_DAT9.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_DATA09 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_DAT9.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_DAT9.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.322 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_ENABLE)

Address: 20E\_0000h base + 518h offset = 20E\_0518h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_ENABLE field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_ENABLE.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_ENABLE.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_ENABLE.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_ENABLE.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_ENABLE.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_ENABLE field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_ENABLE.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_ENABLE.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.323 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_HSYNC)

Address: 20E\_0000h base + 51Ch offset = 20E\_051Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	0									LVE	0						HYS
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_HSYNC field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_HSYNC.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_HSYNC.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_HSYNC.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_HSYNC.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_HSYNC.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_HSYNC field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_HSYNC. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_HSYNC. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.324 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_RESET)

Address: 20E\_0000h base + 520h offset = 20E\_0520h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_RESET field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_RESET.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_RESET.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_RESET.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_RESET.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_RESET.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_RESET field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_RESET. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_RESET. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.325 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_VSYNC)

Address: 20E\_0000h base + 524h offset = 20E\_0524h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		
W																SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_VSYNC field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: LCD_VSYNC.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: LCD_VSYNC.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: LCD_VSYNC.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: LCD_VSYNC.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: LCD_VSYNC.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_LCD\_VSYNC field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: LCD_VSYNC.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: LCD_VSYNC.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.326 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_PWM1)

Address: 20E\_0000h base + 528h offset = 20E\_0528h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_PWM1 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: PWM1.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: PWM1.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: PWM1.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: PWM1.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: PWM1.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_PWM1 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: PWM1. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: PWM1. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.327 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_REF\_CLK\_24M)

Address: 20E\_0000h base + 52Ch offset = 20E\_052Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_REF\_CLK\_24M field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: REF_CLK_24M.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: REF_CLK_24M.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: REF_CLK_24M.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: REF_CLK_24M.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: REF_CLK_24M.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_REF\_CLK\_24M field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: REF_CLK_24M. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: REF_CLK_24M. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.328 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_REF\_CLK\_32K)

Address: 20E\_0000h base + 530h offset = 20E\_0530h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_REF\_CLK\_32K field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: REF_CLK_32K.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: REF_CLK_32K.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: REF_CLK_32K.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: REF_CLK_32K.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: REF_CLK_32K.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_REF\_CLK\_32K field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: REF_CLK_32K.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: REF_CLK_32K.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.329 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_CLK)

Address: 20E\_0000h base + 534h offset = 20E\_0534h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_CLK field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD1_CLK.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD1_CLK.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD1_CLK.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD1_CLK.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD1_CLK.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_CLK field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD1_CLK.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD1_CLK.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.330 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_CMD)

Address: 20E\_0000h base + 538h offset = 20E\_0538h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_CMD field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD1_CMD.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD1_CMD.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD1_CMD.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD1_CMD.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD1_CMD.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_CMD field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD1_CMD. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD1_CMD. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.331 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA0)

Address: 20E\_0000h base + 53Ch offset = 20E\_053Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA0 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD1_DAT0.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD1_DAT0.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD1_DAT0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD1_DAT0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD1_DAT0.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA0 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD1_DAT0.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD1_DAT0.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.332 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA1)

Address: 20E\_0000h base + 540h offset = 20E\_0540h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA1 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD1_DAT1.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD1_DAT1.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD1_DAT1.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD1_DAT1.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD1_DAT1.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA1 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD1_DAT1.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD1_DAT1.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.333 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA2)

Address: 20E\_0000h base + 544h offset = 20E\_0544h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA2 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD1_DAT2.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD1_DAT2.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD1_DAT2.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD1_DAT2.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD1_DAT2.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA2 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD1_DAT2.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD1_DAT2.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.334 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA3)

Address: 20E\_0000h base + 548h offset = 20E\_0548h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA3 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD1_DAT3.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD1_DAT3.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD1_DAT3.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD1_DAT3.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD1_DAT3.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA3 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD1_DAT3.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD1_DAT3.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.335 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA4)

Address: 20E\_0000h base + 54Ch offset = 20E\_054Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA4 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD1_DAT4.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD1_DAT4.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD1_DAT4.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD1_DAT4.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD1_DAT4.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA4 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD1_DAT4.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD1_DAT4.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.336 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA5)

Address: 20E\_0000h base + 550h offset = 20E\_0550h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA5 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD1_DAT5.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD1_DAT5.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD1_DAT5.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD1_DAT5.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD1_DAT5.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA5 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD1_DAT5.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD1_DAT5.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.337 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA6)

Address: 20E\_0000h base + 554h offset = 20E\_0554h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA6 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD1_DAT6.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD1_DAT6.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD1_DAT6.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD1_DAT6.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD1_DAT6.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA6 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD1_DAT6.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD1_DAT6.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.338 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA7)

Address: 20E\_0000h base + 558h offset = 20E\_0558h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA7 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD1_DAT7.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD1_DAT7.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD1_DAT7.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD1_DAT7.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD1_DAT7.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_SD1\_DATA7 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD1_DAT7.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD1_DAT7.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.339 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_CLK)

Address: 20E\_0000h base + 55Ch offset = 20E\_055Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE	0					HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_CLK field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD2_CLK.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD2_CLK.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD2_CLK.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD2_CLK.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD2_CLK.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_CLK field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD2_CLK.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD2_CLK.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.340 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_CMD)

Address: 20E\_0000h base + 560h offset = 20E\_0560h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_CMD field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD2_CMD.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD2_CMD.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD2_CMD.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD2_CMD.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD2_CMD.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_CMD field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD2_CMD.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD2_CMD.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.341 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA0)

Address: 20E\_0000h base + 564h offset = 20E\_0564h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA0 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD2_DAT0.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD2_DAT0.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD2_DAT0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD2_DAT0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD2_DAT0.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA0 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD2_DAT0.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD2_DAT0.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.342 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA1)

Address: 20E\_0000h base + 568h offset = 20E\_0568h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA1 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD2_DAT1.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD2_DAT1.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD2_DAT1.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD2_DAT1.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD2_DAT1.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA1 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD2_DAT1.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD2_DAT1.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.343 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA2)

Address: 20E\_0000h base + 56Ch offset = 20E\_056Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA2 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD2_DAT2.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD2_DAT2.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD2_DAT2.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD2_DAT2.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD2_DAT2.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA2 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD2_DAT2. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD2_DAT2. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.344 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA3)

Address: 20E\_0000h base + 570h offset = 20E\_0570h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA3 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD2_DAT3.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD2_DAT3.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD2_DAT3.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD2_DAT3.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD2_DAT3.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA3 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD2_DAT3.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD2_DAT3.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.345 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA4)

Address: 20E\_0000h base + 574h offset = 20E\_0574h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS	PUE	PKE	ODE	0				SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA4 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD2_DAT4.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD2_DAT4.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD2_DAT4.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD2_DAT4.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD2_DAT4.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA4 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD2_DAT4.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD2_DAT4.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.346 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA5)

Address: 20E\_0000h base + 578h offset = 20E\_0578h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA5 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD2_DAT5.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD2_DAT5.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD2_DAT5.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD2_DAT5.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD2_DAT5.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA5 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD2_DAT5.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD2_DAT5.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.347 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA6)

Address: 20E\_0000h base + 57Ch offset = 20E\_057Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA6 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD2_DAT6.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD2_DAT6.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD2_DAT6.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD2_DAT6.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD2_DAT6.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA6 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD2_DAT6.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD2_DAT6.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.348 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA7)

Address: 20E\_0000h base + 580h offset = 20E\_0580h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA7 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD2_DAT7.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD2_DAT7.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD2_DAT7.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD2_DAT7.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD2_DAT7.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_DATA7 field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD2_DAT7.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD2_DAT7.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.349 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_RESET)

Address: 20E\_0000h base + 584h offset = 20E\_0584h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_RESET field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD2_RST.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD2_RST.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD2_RST.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD2_RST.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD2_RST.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_SD2\_RESET field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD2_RST.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD2_RST.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.350 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_CLK)

Address: 20E\_0000h base + 588h offset = 20E\_0588h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_CLK field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD3_CLK.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD3_CLK.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD3_CLK.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD3_CLK.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD3_CLK.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_CLK field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD3_CLK.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD3_CLK.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.351 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_CMD)

Address: 20E\_0000h base + 58Ch offset = 20E\_058Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE	0					HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS	PUE	PKE	ODE	0				SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_CMD field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD3_CMD.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD3_CMD.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD3_CMD.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD3_CMD.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD3_CMD.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_CMD field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD3_CMD. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD3_CMD. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.352 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_DATA0)

Address: 20E\_0000h base + 590h offset = 20E\_0590h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_DATA0 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD3_DAT0.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD3_DAT0.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD3_DAT0.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD3_DAT0.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD3_DAT0.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_DATA0 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD3_DAT0. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD3_DAT0. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.353 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_DATA1)

Address: 20E\_0000h base + 594h offset = 20E\_0594h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_DATA1 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD3_DAT1.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD3_DAT1.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD3_DAT1.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD3_DAT1.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD3_DAT1.

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_DATA1 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD3_DAT1.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD3_DAT1.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.354 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_DATA2)

Address: 20E\_0000h base + 598h offset = 20E\_0598h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_DATA2 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD3_DAT2.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD3_DAT2.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD3_DAT2.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD3_DAT2.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD3_DAT2.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_DATA2 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD3_DAT2.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD3_DAT2.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate



### 30.4.355 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_DATA3)

Address: 20E\_0000h base + 59Ch offset = 20E\_059Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_DATA3 field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: SD3_DAT3.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: SD3_DAT3.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: SD3_DAT3.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: SD3_DAT3.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: SD3_DAT3.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_SD3\_DATA3 field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: SD3_DAT3.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: SD3_DAT3.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.356 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_UART1\_RXD)

Address: 20E\_0000h base + 5A0h offset = 20E\_05A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_UART1\_RXD field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: UART1_RXD.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: UART1_RXD.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: UART1_RXD.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: UART1_RXD.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: UART1_RXD.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_UART1\_RXD field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: UART1_RXD. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: UART1_RXD. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.357 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_UART1\_TXD)

Address: 20E\_0000h base + 5A4h offset = 20E\_05A4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				
W																HYS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE		PKE	ODE	0		SPEED		DSE		0		SRE	
W																
Reset	0	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_UART1\_TXD field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: UART1_TXD.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: UART1_TXD.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: UART1_TXD.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: UART1_TXD.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: UART1_TXD.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_PAD\_UART1\_TXD field descriptions (continued)**

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin.  0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: UART1_TXD.  00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: UART1_TXD.  000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control.  0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.358 Pad Control Register (IOMUXC\_SW\_PAD\_CTL\_PAD\_WDOG\_B)

Address: 20E\_0000h base + 5A8h offset = 20E\_05A8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0									LVE		0				HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PUS		PUE	PKE	ODE	0			SPEED		DSE			0		SRE
W																
Reset	0	0	0	1	0	0	0	0	1	0	1	1	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_PAD\_WDOG\_B field descriptions

Field	Description
31–23 Reserved	This read-only field is reserved and always has the value 0.
22 LVE	Low Voltage Enable Field Select one of next values for pad: WDOG_B.  0 <b>DISABLED</b> — High Voltage 1 <b>ENABLED</b> — Low Voltage
21–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	Hysteresis Enable Field Select one of next values for pad: WDOG_B.  0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input
15–14 PUS	Pull Up / Down Config. Field Select one of next values for pad: WDOG_B.  00 <b>100K_OHM_PD</b> — 100K Ohm Pull Down 01 <b>47K_OHM_PU</b> — 47K Ohm Pull Up 10 <b>100K_OHM_PU</b> — 100K Ohm Pull Up 11 <b>22K_OHM_PU</b> — 22K Ohm Pull Up
13 PUE	Pull / Keep Select Field Select one of next values for pad: WDOG_B.  0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled
12 PKE	Pull / Keep Enable Field Select one of next values for pad: WDOG_B.

Table continues on the next page...

## IOMUXC\_SW\_PAD\_CTL\_PAD\_WDOG\_B field descriptions (continued)

Field	Description
	0 <b>DISABLED</b> — Pull/Keeper Disabled 1 <b>ENABLED</b> — Pull/Keeper Enabled
11 ODE	Open Drain Enable Field Enables open drain of the pin. 0 <b>DISABLED</b> — Output is CMOS. 1 <b>ENABLED</b> — Output is Open Drain.
10–8 Reserved	This read-only field is reserved and always has the value 0.
7–6 SPEED	Speed Field Select one of next values for pad: WDOG_B. 00 <b>RESERVED0</b> — Reserved 01 <b>50MHZ</b> — Low (50 MHz) 10 <b>100MHZ</b> — Medium (100 MHz) 11 <b>200MHZ</b> — Maximum (200 MHz)
5–3 DSE	Drive Strength Field Select one of next values for pad: WDOG_B. 000 <b>HIZ</b> — HI-Z 001 <b>240_OHM</b> — 240 Ohm 010 <b>120_OHM</b> — 120 Ohm 011 <b>80_OHM</b> — 80 Ohm 100 <b>60_OHM</b> — 60 Ohm 101 <b>48_OHM</b> — 48 Ohm 110 <b>40_OHM</b> — 40 Ohm 111 <b>34_OHM</b> — 34 Ohm
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SRE	Slew Rate Field Slew rate control. 0 <b>SLOW</b> — Slow Slew Rate 1 <b>FAST</b> — Fast Slew Rate

### 30.4.359 Pad Group Control Register (IOMUXC\_SW\_PAD\_CTL\_GRP\_ADDDS)

Address: 20E\_0000h base + 5ACh offset = 20E\_05ACh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																										DSE			0		
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0



**IOMUXC\_SW\_PAD\_CTL\_GRP\_ADDDS field descriptions**

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	<p>Drive Strength Field</p> <p>Select one of next values for group: .</p> <p>Affected pads: DRAM_ADDR00, DRAM_ADDR01, DRAM_ADDR02, DRAM_ADDR03, DRAM_ADDR04, DRAM_ADDR05, DRAM_ADDR06, DRAM_ADDR07, DRAM_ADDR08, DRAM_ADDR09, DRAM_ADDR10, DRAM_ADDR11, DRAM_ADDR12, DRAM_ADDR13, DRAM_ADDR14, DRAM_ADDR15, DRAM_SDBA0, DRAM_SDBA1</p> <p>000 <b>HIZ</b> — HI-Z  001 <b>240_OHM</b> — 240 Ohm  010 <b>120_OHM</b> — 120 Ohm  011 <b>80_OHM</b> — 80 Ohm  100 <b>60_OHM</b> — 60 Ohm  101 <b>48_OHM</b> — 48 Ohm  110 <b>40_OHM</b> — 40 Ohm  111 <b>34_OHM</b> — 34 Ohm</p>
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.360 Pad Group Control Register (IOMUXC\_SW\_PAD\_CTL\_GRP\_DDRMODE\_CTL)

Address: 20E\_0000h base + 5B0h offset = 20E\_05B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0														DDR_INPUT	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_GRP\_DDRMODE\_CTL field descriptions**

Field	Description
31–18 Reserved	This read-only field is reserved and always has the value 0.
17 DDR_INPUT	DDR / CMOS Input Mode Field Select one of next values for group: . Affected pads: DRAM_SDQS0_P, DRAM_SDQS1_P, DRAM_SDQS2_P, DRAM_SDQS3_P  0 <b>CMOS</b> — CMOS input mode. 1 <b>DIFFERENTIAL</b> — Differential input mode.
16–0 Reserved	This read-only field is reserved and always has the value 0.

**30.4.361 Pad Group Control Register (IOMUXC\_SW\_PAD\_CTL\_GRP\_DDRPKE)**

Address: 20E\_0000h base + 5B4h offset = 20E\_05B4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			PKE	0											
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_GRP\_DDRPKE field descriptions**

Field	Description
31–13 Reserved	This read-only field is reserved and always has the value 0.
12 PKE	Pull / Keep Enable Field Select one of next values for group: . Affected pads: DRAM_ADDR00, DRAM_ADDR01, DRAM_ADDR02, DRAM_ADDR03, DRAM_ADDR04, DRAM_ADDR05, DRAM_ADDR06, DRAM_ADDR07, DRAM_ADDR08, DRAM_ADDR09, DRAM_ADDR10, DRAM_ADDR11, DRAM_ADDR12, DRAM_ADDR13, DRAM_ADDR14, DRAM_ADDR15, DRAM_CAS_B, DRAM_CS0_B, DRAM_CS1_B, DRAM_DATA00, DRAM_DATA01, DRAM_DATA02, DRAM_DATA03, DRAM_DATA04, DRAM_DATA05, DRAM_DATA06, DRAM_DATA07, DRAM_DATA08, DRAM_DATA09, DRAM_DATA10, DRAM_DATA11, DRAM_DATA12, DRAM_DATA13, DRAM_DATA14, DRAM_DATA15, DRAM_DATA16, DRAM_DATA17, DRAM_DATA18, DRAM_DATA19, DRAM_DATA20, DRAM_DATA21, DRAM_DATA22, DRAM_DATA23, DRAM_DATA24, DRAM_DATA25, DRAM_DATA26, DRAM_DATA27, DRAM_DATA28, DRAM_DATA29, DRAM_DATA30, DRAM_DATA31, DRAM_DQM0, DRAM_DQM1, DRAM_DQM2, DRAM_DQM3, DRAM_RAS_B, DRAM_SDBA0, DRAM_SDBA1, DRAM_SDCLK0_P, DRAM_SDWE_B

*Table continues on the next page...*

**IOMUXC\_SW\_PAD\_CTL\_GRP\_DDRPKE field descriptions (continued)**

Field	Description
0	<b>DISABLED</b> — Pull/Keeper Disabled
1	<b>ENABLED</b> — Pull/Keeper Enabled
11–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.362 Pad Group Control Register (IOMUXC\_SW\_PAD\_CTL\_GRP\_DDRPK)

Address: 20E\_0000h base + 5B8h offset = 20E\_05B8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		PUE	0												
W																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_GRP\_DDRPK field descriptions**

Field	Description
31–14 Reserved	This read-only field is reserved and always has the value 0.
13 PUE	<p>Pull / Keep Select Field</p> <p>Select one of next values for group: .</p> <p>Affected pads: DRAM_ADDR00, DRAM_ADDR01, DRAM_ADDR02, DRAM_ADDR03, DRAM_ADDR04, DRAM_ADDR05, DRAM_ADDR06, DRAM_ADDR07, DRAM_ADDR08, DRAM_ADDR09, DRAM_ADDR10, DRAM_ADDR11, DRAM_ADDR12, DRAM_ADDR13, DRAM_ADDR14, DRAM_ADDR15, DRAM_CAS_B, DRAM_CS0_B, DRAM_CS1_B, DRAM_DATA00, DRAM_DATA01, DRAM_DATA02, DRAM_DATA03, DRAM_DATA04, DRAM_DATA05, DRAM_DATA06, DRAM_DATA07, DRAM_DATA08, DRAM_DATA09, DRAM_DATA10, DRAM_DATA11, DRAM_DATA12, DRAM_DATA13, DRAM_DATA14, DRAM_DATA15, DRAM_DATA16, DRAM_DATA17, DRAM_DATA18, DRAM_DATA19, DRAM_DATA20, DRAM_DATA21, DRAM_DATA22, DRAM_DATA23, DRAM_DATA24, DRAM_DATA25, DRAM_DATA26, DRAM_DATA27, DRAM_DATA28, DRAM_DATA29, DRAM_DATA30, DRAM_DATA31, DRAM_DQM0, DRAM_DQM1, DRAM_DQM2, DRAM_DQM3, DRAM_RAS_B, DRAM_SDBA0, DRAM_SDBA1, DRAM_SDCLK0_P, DRAM_SDWE_B</p> <p>0 <b>KEEP</b> — Keeper Enabled 1 <b>PULL</b> — Pull Enabled</p>
12–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.363 Pad Group Control Register (IOMUXC\_SW\_PAD\_CTL\_GRP\_DDRHYS)

Address: 20E\_0000h base + 5BCh offset = 20E\_05BCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															HYS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_GRP\_DDRHYS field descriptions

Field	Description
31–17 Reserved	This read-only field is reserved and always has the value 0.
16 HYS	<p>Hysteresis Enable Field</p> <p>Select one of next values for group: .</p> <p>Affected pads: DRAM_DATA00, DRAM_DATA01, DRAM_DATA02, DRAM_DATA03, DRAM_DATA04, DRAM_DATA05, DRAM_DATA06, DRAM_DATA07, DRAM_DATA08, DRAM_DATA09, DRAM_DATA10, DRAM_DATA11, DRAM_DATA12, DRAM_DATA13, DRAM_DATA14, DRAM_DATA15, DRAM_DATA16, DRAM_DATA17, DRAM_DATA18, DRAM_DATA19, DRAM_DATA20, DRAM_DATA21, DRAM_DATA22, DRAM_DATA23, DRAM_DATA24, DRAM_DATA25, DRAM_DATA26, DRAM_DATA27, DRAM_DATA28, DRAM_DATA29, DRAM_DATA30, DRAM_DATA31, DRAM_SDQS0_P, DRAM_SDQS1_P, DRAM_SDQS2_P, DRAM_SDQS3_P</p> <p>0 <b>DISABLED</b> — CMOS input 1 <b>ENABLED</b> — Schmitt trigger input</p>
15–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.364 Pad Group Control Register (IOMUXC\_SW\_PAD\_CTL\_GRP\_DDRMODE)

Address: 20E\_0000h base + 5C0h offset = 20E\_05C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0														DDR_INPUT	0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_SW\_PAD\_CTL\_GRP\_DDRMODE field descriptions

Field	Description
31–18 Reserved	This read-only field is reserved and always has the value 0.
17 DDR_INPUT	<p>DDR / CMOS Input Mode Field</p> <p>Select one of next values for group: .</p> <p>Affected pads: DRAM_DATA00, DRAM_DATA01, DRAM_DATA02, DRAM_DATA03, DRAM_DATA04, DRAM_DATA05, DRAM_DATA06, DRAM_DATA07, DRAM_DATA08, DRAM_DATA09, DRAM_DATA10, DRAM_DATA11, DRAM_DATA12, DRAM_DATA13, DRAM_DATA14, DRAM_DATA15, DRAM_DATA16, DRAM_DATA17, DRAM_DATA18, DRAM_DATA19, DRAM_DATA20, DRAM_DATA21, DRAM_DATA22, DRAM_DATA23, DRAM_DATA24, DRAM_DATA25, DRAM_DATA26, DRAM_DATA27, DRAM_DATA28, DRAM_DATA29, DRAM_DATA30, DRAM_DATA31</p> <p>0 <b>CMOS</b> — CMOS input mode.</p> <p>1 <b>DIFFERENTIAL</b> — Differential input mode.</p>
16–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.365 Pad Group Control Register (IOMUXC\_SW\_PAD\_CTL\_GRP\_B0DS)

Address: 20E\_0000h base + 5C4h offset = 20E\_05C4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																DSE								0							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	

#### IOMUXC\_SW\_PAD\_CTL\_GRP\_B0DS field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	<p>Drive Strength Field</p> <p>Select one of next values for group: .</p> <p>Affected pads: DRAM_DATA00, DRAM_DATA01, DRAM_DATA02, DRAM_DATA03, DRAM_DATA04, DRAM_DATA05, DRAM_DATA06, DRAM_DATA07</p> <p>000 <b>HIZ</b> — HI-Z</p> <p>001 <b>240_OHM</b> — 240 Ohm</p> <p>010 <b>120_OHM</b> — 120 Ohm</p> <p>011 <b>80_OHM</b> — 80 Ohm</p> <p>100 <b>60_OHM</b> — 60 Ohm</p> <p>101 <b>48_OHM</b> — 48 Ohm</p> <p>110 <b>40_OHM</b> — 40 Ohm</p> <p>111 <b>34_OHM</b> — 34 Ohm</p>
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.366 Pad Group Control Register (IOMUXC\_SW\_PAD\_CTL\_GRP\_CTLDS)

Address: 20E\_0000h base + 5C8h offset = 20E\_05C8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																DSE								0							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	

#### IOMUXC\_SW\_PAD\_CTL\_GRP\_CTLDS field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_GRP\_CTLDS field descriptions (continued)**

Field	Description
5–3 DSE	<p>Drive Strength Field</p> <p>Select one of next values for group: .</p> <p>Affected pads: DRAM_CS0_B, DRAM_CS1_B, DRAM_SDBA2, DRAM_SDCKE0, DRAM_SDCKE1, DRAM_SDWE_B</p> <p>000 <b>HIZ</b> — HI-Z</p> <p>001 <b>240_OHM</b> — 240 Ohm</p> <p>010 <b>120_OHM</b> — 120 Ohm</p> <p>011 <b>80_OHM</b> — 80 Ohm</p> <p>100 <b>60_OHM</b> — 60 Ohm</p> <p>101 <b>48_OHM</b> — 48 Ohm</p> <p>110 <b>40_OHM</b> — 40 Ohm</p> <p>111 <b>34_OHM</b> — 34 Ohm</p>
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.367 Pad Group Control Register (IOMUXC\_SW\_PAD\_CTL\_GRP\_B1DS)

Address: 20E\_0000h base + 5CCCh offset = 20E\_05CCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																DSE										0					
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0

**IOMUXC\_SW\_PAD\_CTL\_GRP\_B1DS field descriptions**

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	<p>Drive Strength Field</p> <p>Select one of next values for group: .</p> <p>Affected pads: DRAM_DATA08, DRAM_DATA09, DRAM_DATA10, DRAM_DATA11, DRAM_DATA12, DRAM_DATA13, DRAM_DATA14, DRAM_DATA15</p> <p>000 <b>HIZ</b> — HI-Z</p> <p>001 <b>240_OHM</b> — 240 Ohm</p> <p>010 <b>120_OHM</b> — 120 Ohm</p> <p>011 <b>80_OHM</b> — 80 Ohm</p> <p>100 <b>60_OHM</b> — 60 Ohm</p> <p>101 <b>48_OHM</b> — 48 Ohm</p> <p>110 <b>40_OHM</b> — 40 Ohm</p> <p>111 <b>34_OHM</b> — 34 Ohm</p>

*Table continues on the next page...*

## IOMUXC\_SW\_PAD\_CTL\_GRP\_B1DS field descriptions (continued)

Field	Description
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.368 Pad Group Control Register (IOMUXC\_SW\_PAD\_CTL\_GRP\_DDR\_TYPE)

Address: 20E\_0000h base + 5D0h offset = 20E\_05D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												DDR_SEL		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## IOMUXC\_SW\_PAD\_CTL\_GRP\_DDR\_TYPE field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 DDR_SEL	<p>DDR Select Field</p> <p>Select one of next values for group: .</p> <p>Affected pads: DRAM_ADDR00, DRAM_ADDR01, DRAM_ADDR02, DRAM_ADDR03, DRAM_ADDR04, DRAM_ADDR05, DRAM_ADDR06, DRAM_ADDR07, DRAM_ADDR08, DRAM_ADDR09, DRAM_ADDR10, DRAM_ADDR11, DRAM_ADDR12, DRAM_ADDR13, DRAM_ADDR14, DRAM_ADDR15, DRAM_CAS_B, DRAM_CS0_B, DRAM_CS1_B, DRAM_DATA00, DRAM_DATA01, DRAM_DATA02, DRAM_DATA03, DRAM_DATA04, DRAM_DATA05, DRAM_DATA06, DRAM_DATA07, DRAM_DATA08, DRAM_DATA09, DRAM_DATA10, DRAM_DATA11, DRAM_DATA12, DRAM_DATA13, DRAM_DATA14, DRAM_DATA15, DRAM_DATA16, DRAM_DATA17, DRAM_DATA18, DRAM_DATA19, DRAM_DATA20, DRAM_DATA21, DRAM_DATA22, DRAM_DATA23, DRAM_DATA24, DRAM_DATA25, DRAM_DATA26, DRAM_DATA27, DRAM_DATA28, DRAM_DATA29, DRAM_DATA30, DRAM_DATA31, DRAM_DQM0, DRAM_DQM1, DRAM_DQM2, DRAM_DQM3, DRAM_ODT0, DRAM_ODT1, DRAM_RAS_B, DRAM_SDBA0, DRAM_SDBA1, DRAM_SDBA2, DRAM_SDCKE0, DRAM_SDCKE1, DRAM_SDCLK0_P, DRAM_SDQS0_P, DRAM_SDQS1_P, DRAM_SDQS2_P, DRAM_SDQS3_P, DRAM_SDWE_B</p> <p>00 <b>RESERVED0</b> — Reserved</p> <p>01 <b>RESERVED1</b> — Reserved</p> <p>10 <b>LPDDR2</b> — LPDDR2 mode (240 Ohm driver unit calibration, 240, 120, 80, 60, 48, 40, 32 Ohm drive strengths at 1.2V)</p> <p>11 <b>DDR3</b> — DDR3 mode (240 Ohm driver unit calibration, 240, 120, 80, 60, 48, 40, 32 Ohm drive strengths at 1.5V)</p>
17–0 Reserved	This read-only field is reserved and always has the value 0.



### 30.4.369 Pad Group Control Register (IOMUXC\_SW\_PAD\_CTL\_GRP\_B2DS)

Address: 20E\_0000h base + 5D4h offset = 20E\_05D4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																DSE				0											
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	

#### IOMUXC\_SW\_PAD\_CTL\_GRP\_B2DS field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value 0.
5–3 DSE	<p>Drive Strength Field</p> <p>Select one of next values for group: .</p> <p>Affected pads: DRAM_DATA16, DRAM_DATA17, DRAM_DATA18, DRAM_DATA19, DRAM_DATA20, DRAM_DATA21, DRAM_DATA22, DRAM_DATA23</p> <p>000 <b>HIZ</b> — HI-Z</p> <p>001 <b>240_OHM</b> — 240 Ohm</p> <p>010 <b>120_OHM</b> — 120 Ohm</p> <p>011 <b>80_OHM</b> — 80 Ohm</p> <p>100 <b>60_OHM</b> — 60 Ohm</p> <p>101 <b>48_OHM</b> — 48 Ohm</p> <p>110 <b>40_OHM</b> — 40 Ohm</p> <p>111 <b>34_OHM</b> — 34 Ohm</p>
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.370 Pad Group Control Register (IOMUXC\_SW\_PAD\_CTL\_GRP\_B3DS)

Address: 20E\_0000h base + 5D8h offset = 20E\_05D8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																DSE				0											
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	

#### IOMUXC\_SW\_PAD\_CTL\_GRP\_B3DS field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**IOMUXC\_SW\_PAD\_CTL\_GRP\_B3DS field descriptions (continued)**

Field	Description
5–3 DSE	<p>Drive Strength Field</p> <p>Select one of next values for group: .</p> <p>Affected pads: DRAM_DATA24, DRAM_DATA25, DRAM_DATA26, DRAM_DATA27, DRAM_DATA28, DRAM_DATA29, DRAM_DATA30, DRAM_DATA31</p> <p>000 <b>HIZ</b> — HI-Z</p> <p>001 <b>240_OHM</b> — 240 Ohm</p> <p>010 <b>120_OHM</b> — 120 Ohm</p> <p>011 <b>80_OHM</b> — 80 Ohm</p> <p>100 <b>60_OHM</b> — 60 Ohm</p> <p>101 <b>48_OHM</b> — 48 Ohm</p> <p>110 <b>40_OHM</b> — 40 Ohm</p> <p>111 <b>34_OHM</b> — 34 Ohm</p>
2–0 Reserved	This read-only field is reserved and always has the value 0.

### 30.4.371 Select Input Register (IOMUXC\_ANALOG\_USB\_OTG\_ID\_SELECT\_INPUT)

Address: 20E\_0000h base + 5DCCh offset = 20E\_05DCCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																	0															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_ANALOG\_USB\_OTG\_ID\_SELECT\_INPUT field descriptions**

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value 0.
2–0 DAISY	<p>Input Select (DAISY) Field</p> <p>Selecting Pads Involved in Daisy Chain.</p> <p>000 <b>EPDC_PWR_COM_ALT4</b> — Selecting ALT4 mode of pad EPDC_PWRCOM for USB_OTG1_ID.</p> <p>001 <b>FEC_RX_DATA0_ALT2</b> — Selecting ALT2 mode of pad FEC_RXD0 for USB_OTG1_ID.</p> <p>010 <b>LCD_DATA01_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT1 for USB_OTG1_ID.</p> <p>011 <b>REF_CLK_32K_ALT3</b> — Selecting ALT3 mode of pad REF_CLK_32K for USB_OTG1_ID.</p> <p>100 <b>SD3_DATA0_ALT4</b> — Selecting ALT4 mode of pad SD3_DAT0 for USB_OTG1_ID.</p>

### 30.4.372 Select Input Register (IOMUXC\_ANALOG\_USB\_H1\_ID\_SELECT\_INPUT)

Address: 20E\_0000h base + 5E0h offset = 20E\_05E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_ANALOG\_USB\_H1\_ID\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_PWR_IRQ_ALT4</b> — Selecting ALT4 mode of pad EPDC_PWRINT for USB_OTG2_ID. 01 <b>LCD_DATA00_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT0 for USB_OTG2_ID. 10 <b>REF_CLK_24M_ALT3</b> — Selecting ALT3 mode of pad REF_CLK_24M for USB_OTG2_ID. 11 <b>SD3_CMD_ALT4</b> — Selecting ALT4 mode of pad SD3_CMD for USB_OTG2_ID.

### 30.4.373 Select Input Register (IOMUXC\_AUD4\_INPUT\_DA\_AMX\_SELECT\_INPUT)

Address: 20E\_0000h base + 5E4h offset = 20E\_05E4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_AUD4\_INPUT\_DA\_AMX\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>ECSPI1_SS0_ALT1</b> — Selecting ALT1 mode of pad ECSPI1_SS0 for AUD4_RXD. 01 <b>LCD_DATA03_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT3 for AUD4_RXD. 10 <b>SD2_DATA0_ALT1</b> — Selecting ALT1 mode of pad SD2_DAT0 for AUD4_RXD.

### 30.4.374 Select Input Register (IOMUXC\_AUD4\_INPUT\_DB\_AMX\_SELECT\_INPUT)

Address: 20E\_0000h base + 5E8h offset = 20E\_05E8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_AUD4\_INPUT\_DB\_AMX\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>ECSPI1_SCLK_ALT1</b> — Selecting ALT1 mode of pad ECSPI1_SCLK for AUD4_TXD. 01 <b>LCD_DATA06_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT6 for AUD4_TXD. 10 <b>SD2_DATA3_ALT1</b> — Selecting ALT1 mode of pad SD2_DAT3 for AUD4_TXD.

### 30.4.375 Select Input Register (IOMUXC\_AUD4\_INPUT\_RXCLK\_AMX\_SELECT\_INPUT)

Address: 20E\_0000h base + 5ECh offset = 20E\_05ECh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_AUD4\_INPUT\_RXCLK\_AMX\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>I2C2_SDA_ALT1</b> — Selecting ALT1 mode of pad I2C2_SDA for AUD4_RXC. 01 <b>LCD_DATA02_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT2 for AUD4_RXC. 10 <b>SD2_CMD_ALT1</b> — Selecting ALT1 mode of pad SD2_CMD for AUD4_RXC.

### 30.4.376 Select Input Register (IOMUXC\_AUD4\_INPUT\_RXFS\_AMX\_SELECT\_INPUT)

Address: 20E\_0000h base + 5F0h offset = 20E\_05F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_AUD4\_INPUT\_RXFS\_AMX\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**IOMUXC\_AUD4\_INPUT\_RXFS\_AMX\_SELECT\_INPUT field descriptions (continued)**

Field	Description
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>I2C2_SCL_ALT1</b> — Selecting ALT1 mode of pad I2C2_SCL for AUD4_RXFS. 01 <b>LCD_DATA01_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT1 for AUD4_RXFS. 10 <b>SD2_CLK_ALT1</b> — Selecting ALT1 mode of pad SD2_CLK for AUD4_RXFS.

### 30.4.377 Select Input Register (IOMUXC\_AUD4\_INPUT\_TXCLK\_AMX\_SELECT\_INPUT)

Address: 20E\_0000h base + 5F4h offset = 20E\_05F4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_AUD4\_INPUT\_TXCLK\_AMX\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>ECSPI1_MOSI_ALT1</b> — Selecting ALT1 mode of pad ECSPI1_MOSI for AUD4_TXC. 01 <b>LCD_DATA04_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT4 for AUD4_TXC. 10 <b>SD2_DATA1_ALT1</b> — Selecting ALT1 mode of pad SD2_DAT1 for AUD4_TXC.

### 30.4.378 Select Input Register (IOMUXC\_AUD4\_INPUT\_TXFS\_AMX\_SELECT\_INPUT)

Address: 20E\_0000h base + 5F8h offset = 20E\_05F8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_AUD4\_INPUT\_TXFS\_AMX\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>ECSPI1_MISO_ALT1</b> — Selecting ALT1 mode of pad ECSPI1_MISO for AUD4_TXFS. 01 <b>LCD_DATA05_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT5 for AUD4_TXFS. 10 <b>SD2_DATA2_ALT1</b> — Selecting ALT1 mode of pad SD2_DAT2 for AUD4_TXFS.

### 30.4.379 Select Input Register (IOMUXC\_AUD5\_INPUT\_DA\_AMX\_SELECT\_INPUT)

Address: 20E\_0000h base + 5FCh offset = 20E\_05FCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_AUD5\_INPUT\_DA\_AMX\_SELECT\_INPUT field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_VCOM1_ALT1</b> — Selecting ALT1 mode of pad EPDC_VCOM1 for AUD5_RXD. 1 <b>SD3_DATA0_ALT1</b> — Selecting ALT1 mode of pad SD3_DAT0 for AUD5_RXD.

### 30.4.380 Select Input Register (IOMUXC\_AUD5\_INPUT\_DB\_AMX\_SELECT\_INPUT)

Address: 20E\_0000h base + 600h offset = 20E\_0600h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_AUD5\_INPUT\_DB\_AMX\_SELECT\_INPUT field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_PWR_CTRL2_ALT1</b> — Selecting ALT1 mode of pad EPDC_PWRCTRL2 for AUD5_TXD. 1 <b>SD3_DATA3_ALT1</b> — Selecting ALT1 mode of pad SD3_DAT3 for AUD5_TXD.



### 30.4.381 Select Input Register (IOMUXC\_AUD5\_INPUT\_RXCLK\_AMX\_SELECT\_INPUT)

Address: 20E\_0000h base + 604h offset = 20E\_0604h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_AUD5\_INPUT\_RXCLK\_AMX\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_PWR_CTRL0_ALT1</b> — Selecting ALT1 mode of pad EPDC_PWRCTRL0 for AUD5_RXC. 1 <b>SD3_CMD_ALT1</b> — Selecting ALT1 mode of pad SD3_CMD for AUD5_RXC.

30.4.382 Select Input Register  
(IOMUXC\_AUD5\_INPUT\_RXFS\_AMX\_SELECT\_INPUT)

Address: 20E\_0000h base + 608h offset = 20E\_0608h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_AUD5\_INPUT\_RXFS\_AMX\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_VCOM0_ALT1</b> — Selecting ALT1 mode of pad EPDC_VCOM0 for AUD5_RXFS. 1 <b>SD3_CLK_ALT1</b> — Selecting ALT1 mode of pad SD3_CLK for AUD5_RXFS.

### 30.4.383 Select Input Register (IOMUXC\_AUD5\_INPUT\_TXCLK\_AMX\_SELECT\_INPUT)

Address: 20E\_0000h base + 60Ch offset = 20E\_060Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_AUD5\_INPUT\_TXCLK\_AMX\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_PWR_CTRL3_ALT1</b> — Selecting ALT1 mode of pad EPDC_PWRCTRL3 for AUD5_TXC. 1 <b>SD3_DATA1_ALT1</b> — Selecting ALT1 mode of pad SD3_DAT1 for AUD5_TXC.

30.4.384    **Select Input Register**  
**(IOMUXC\_AUD5\_INPUT\_TXFS\_AMX\_SELECT\_INPUT)**

Address: 20E\_0000h base + 610h offset = 20E\_0610h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_AUD5\_INPUT\_TXFS\_AMX\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_PWR_CTRL1_ALT1</b> — Selecting ALT1 mode of pad EPDC_PWRCTRL1 for AUD5_TXFS. 1 <b>SD3_DATA2_ALT1</b> — Selecting ALT1 mode of pad SD3_DAT2 for AUD5_TXFS.

30.4.385 Select Input Register  
(IOMUXC\_AUD6\_INPUT\_DA\_AMX\_SELECT\_INPUT)

Address: 20E\_0000h base + 614h offset = 20E\_0614h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_AUD6\_INPUT\_DA\_AMX\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>FEC_RX_ER_ALT2</b> — Selecting ALT2 mode of pad FEC_RX_ER for AUD6_RXD. 1 <b>KEY_COL4_ALT1</b> — Selecting ALT1 mode of pad KEY_COL4 for AUD6_RXD.

30.4.386 Select Input Register  
(IOMUXC\_AUD6\_INPUT\_DB\_AMX\_SELECT\_INPUT)

Address: 20E\_0000h base + 618h offset = 20E\_0618h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_AUD6\_INPUT\_DB\_AMX\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>FEC_TX_DATA0_ALT2</b> — Selecting ALT2 mode of pad FEC_TXD0 for AUD6_TXD. 1 <b>KEY_ROW5_ALT1</b> — Selecting ALT1 mode of pad KEY_ROW5 for AUD6_TXD.

### 30.4.387 Select Input Register (IOMUXC\_AUD6\_INPUT\_RXCLK\_AMX\_SELECT\_INPUT)

Address: 20E\_0000h base + 61Ch offset = 20E\_061Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_AUD6\_INPUT\_RXCLK\_AMX\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>FEC_TX_CLK_ALT2</b> — Selecting ALT2 mode of pad FEC_TX_CLK for AUD6_RXC. 1 <b>KEY_ROW3_ALT1</b> — Selecting ALT1 mode of pad KEY_ROW3 for AUD6_RXC.

### 30.4.388 Select Input Register (IOMUXC\_AUD6\_INPUT\_RXFS\_AMX\_SELECT\_INPUT)

Address: 20E\_0000h base + 620h offset = 20E\_0620h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_AUD6\_INPUT\_RXFS\_AMX\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>FEC_MDIO_ALT2</b> — Selecting ALT2 mode of pad FEC_MDIO for AUD6_RXFS. 1 <b>KEY_COL3_ALT1</b> — Selecting ALT1 mode of pad KEY_COL3 for AUD6_RXFS.



### 30.4.389 Select Input Register (IOMUXC\_AUD6\_INPUT\_TXCLK\_AMX\_SELECT\_INPUT)

Address: 20E\_0000h base + 624h offset = 20E\_0624h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_AUD6\_INPUT\_TXCLK\_AMX\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>FEC_CRS_DV_ALT2</b> — Selecting ALT2 mode of pad FEC_CRS_DV for AUD6_TXC. 1 <b>KEY_ROW4_ALT1</b> — Selecting ALT1 mode of pad KEY_ROW4 for AUD6_TXC.

### 30.4.390 Select Input Register (IOMUXC\_AUD6\_INPUT\_TXFS\_AMX\_SELECT\_INPUT)

Address: 20E\_0000h base + 628h offset = 20E\_0628h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_AUD6\_INPUT\_TXFS\_AMX\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>FEC_RX_DATA1_ALT2</b> — Selecting ALT2 mode of pad FEC_RXD1 for AUD6_TXFS. 1 <b>KEY_COL5_ALT1</b> — Selecting ALT1 mode of pad KEY_COL5 for AUD6_TXFS.

### 30.4.391 Select Input Register (IOMUXC\_CCM\_PMIC\_READY\_SELECT\_INPUT)

Address: 20E\_0000h base + 62Ch offset = 20E\_062Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_CCM\_PMIC\_READY\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>FEC_REF_CLK_ALT4</b> — Selecting ALT4 mode of pad FEC_REF_CLK for CCM_PMIC_READY. 01 <b>LCD_RESET_ALT6</b> — Selecting ALT6 mode of pad LCD_RESET for CCM_PMIC_READY. 10 <b>REF_CLK_24M_ALT4</b> — Selecting ALT4 mode of pad REF_CLK_24M for CCM_PMIC_READY. 11 <b>SD1_DATA7_ALT3</b> — Selecting ALT3 mode of pad SD1_DAT7 for CCM_PMIC_READY.

### 30.4.392 Select Input Register (IOMUXC\_CSI\_CSI\_DATA00\_SELECT\_INPUT)

Address: 20E\_0000h base + 630h offset = 20E\_0630h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_CSI\_CSI\_DATA00\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_DATA00_ALT3</b> — Selecting ALT3 mode of pad EPDC_D0 for CSI_DATA00. 01 <b>LCD_DATA17_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT17 for CSI_DATA00. 10 <b>SD2_CLK_ALT3</b> — Selecting ALT3 mode of pad SD2_CLK for CSI_DATA00.

### 30.4.393 Select Input Register (IOMUXC\_CSI\_CSI\_DATA01\_SELECT\_INPUT)

Address: 20E\_0000h base + 634h offset = 20E\_0634h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_CSI\_CSI\_DATA01\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_DATA01_ALT3</b> — Selecting ALT3 mode of pad EPDC_D1 for CSI_DATA01. 01 <b>LCD_DATA16_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT16 for CSI_DATA01. 10 <b>SD2_CMD_ALT3</b> — Selecting ALT3 mode of pad SD2_CMD for CSI_DATA01.

### 30.4.394 Select Input Register (IOMUXC\_CSI\_CSI\_DATA02\_SELECT\_INPUT)

Address: 20E\_0000h base + 638h offset = 20E\_0638h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_CSI\_CSI\_DATA02\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**IOMUXC\_CSI\_CSI\_DATA02\_SELECT\_INPUT field descriptions (continued)**

Field	Description
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_DATA02_ALT3</b> — Selecting ALT3 mode of pad EPDC_D2 for CSI_DATA02. 01 <b>LCD_DATA15_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT15 for CSI_DATA02. 10 <b>SD2_DATA0_ALT3</b> — Selecting ALT3 mode of pad SD2_DAT0 for CSI_DATA02.

### 30.4.395 Select Input Register (IOMUXC\_CSI\_CSI\_DATA03\_SELECT\_INPUT)

Address: 20E\_0000h base + 63Ch offset = 20E\_063Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_CSI\_CSI\_DATA03\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_DATA03_ALT3</b> — Selecting ALT3 mode of pad EPDC_D3 for CSI_DATA03. 01 <b>LCD_DATA14_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT14 for CSI_DATA03. 10 <b>SD2_DATA1_ALT3</b> — Selecting ALT3 mode of pad SD2_DAT1 for CSI_DATA03.

### 30.4.396 Select Input Register (IOMUXC\_CSI\_CSI\_DATA04\_SELECT\_INPUT)

Address: 20E\_0000h base + 640h offset = 20E\_0640h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_CSI\_CSI\_DATA04\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_DATA04_ALT3</b> — Selecting ALT3 mode of pad EPDC_D4 for CSI_DATA04. 01 <b>LCD_DATA13_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT13 for CSI_DATA04. 10 <b>SD2_DATA2_ALT3</b> — Selecting ALT3 mode of pad SD2_DAT2 for CSI_DATA04.

### 30.4.397 Select Input Register (IOMUXC\_CSI\_CSI\_DATA05\_SELECT\_INPUT)

Address: 20E\_0000h base + 644h offset = 20E\_0644h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_CSI\_CSI\_DATA05\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**IOMUXC\_CSI\_CSI\_DATA05\_SELECT\_INPUT field descriptions (continued)**

Field	Description
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_DATA05_ALT3</b> — Selecting ALT3 mode of pad EPDC_D5 for CSI_DATA05. 01 <b>LCD_DATA12_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT12 for CSI_DATA05. 10 <b>SD2_DATA3_ALT3</b> — Selecting ALT3 mode of pad SD2_DAT3 for CSI_DATA05.

### 30.4.398 Select Input Register (IOMUXC\_CSI\_CSI\_DATA06\_SELECT\_INPUT)

Address: 20E\_0000h base + 648h offset = 20E\_0648h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_CSI\_CSI\_DATA06\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_DATA06_ALT3</b> — Selecting ALT3 mode of pad EPDC_D6 for CSI_DATA06. 01 <b>LCD_DATA11_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT11 for CSI_DATA06. 10 <b>SD2_DATA4_ALT3</b> — Selecting ALT3 mode of pad SD2_DAT4 for CSI_DATA06.

### 30.4.399 Select Input Register (IOMUXC\_CSI\_CSI\_DATA07\_SELECT\_INPUT)

Address: 20E\_0000h base + 64Ch offset = 20E\_064Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_CSI\_CSI\_DATA07\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_DATA07_ALT3</b> — Selecting ALT3 mode of pad EPDC_D7 for CSI_DATA07. 01 <b>LCD_DATA10_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT10 for CSI_DATA07. 10 <b>SD2_DATA5_ALT3</b> — Selecting ALT3 mode of pad SD2_DAT5 for CSI_DATA07.

### 30.4.400 Select Input Register (IOMUXC\_CSI\_CSI\_DATA08\_SELECT\_INPUT)

Address: 20E\_0000h base + 650h offset = 20E\_0650h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_CSI\_CSI\_DATA08\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...



**IOMUXC\_CSI\_CSI\_DATA08\_SELECT\_INPUT field descriptions (continued)**

Field	Description
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_SDCLK_ALT3</b> — Selecting ALT3 mode of pad EPDC_SDCLK for CSI_DATA08. 01 <b>LCD_DATA09_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT9 for CSI_DATA08. 10 <b>SD2_DATA6_ALT3</b> — Selecting ALT3 mode of pad SD2_DAT6 for CSI_DATA08.

### 30.4.401 Select Input Register (IOMUXC\_CSI\_CSI\_DATA09\_SELECT\_INPUT)

Address: 20E\_0000h base + 654h offset = 20E\_0654h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_CSI\_CSI\_DATA09\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_SDLE_ALT3</b> — Selecting ALT3 mode of pad EPDC_SDLE for CSI_DATA09. 01 <b>LCD_DATA08_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT8 for CSI_DATA09. 10 <b>SD2_DATA7_ALT3</b> — Selecting ALT3 mode of pad SD2_DAT7 for CSI_DATA09.

### 30.4.402 Select Input Register (IOMUXC\_CSI\_CSI\_DATA10\_SELECT\_INPUT)

Address: 20E\_0000h base + 658h offset = 20E\_0658h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_CSI\_CSI\_DATA10\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_SDOE_ALT3</b> — Selecting ALT3 mode of pad EPDC_SDOE for CSI_DATA10. 01 <b>LCD_DATA23_ALT2</b> — Selecting ALT2 mode of pad LCD_DATA23 for CSI_DATA10. 10 <b>SD3_CLK_ALT3</b> — Selecting ALT3 mode of pad SD3_CLK for CSI_DATA10.

### 30.4.403 Select Input Register (IOMUXC\_CSI\_CSI\_DATA11\_SELECT\_INPUT)

Address: 20E\_0000h base + 65Ch offset = 20E\_065Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_CSI\_CSI\_DATA11\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**IOMUXC\_CSI\_CSI\_DATA11\_SELECT\_INPUT field descriptions (continued)**

Field	Description
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_SDSHR_ALT3</b> — Selecting ALT3 mode of pad EPDC_SDSHR for CSI_DATA11. 01 <b>LCD_DATA22_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT22 for CSI_DATA11. 10 <b>SD3_CMD_ALT3</b> — Selecting ALT3 mode of pad SD3_CMD for CSI_DATA11.

### 30.4.404 Select Input Register (IOMUXC\_CSI\_CSI\_DATA12\_SELECT\_INPUT)

Address: 20E\_0000h base + 660h offset = 20E\_0660h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_CSI\_CSI\_DATA12\_SELECT\_INPUT field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>LCD_DATA21_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT21 for CSI_DATA12. 1 <b>SD3_DATA0_ALT3</b> — Selecting ALT3 mode of pad SD3_DAT0 for CSI_DATA12.

30.4.405 Select Input Register  
(IOMUXC\_CSI\_CSI\_DATA13\_SELECT\_INPUT)

Address: 20E\_0000h base + 664h offset = 20E\_0664h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

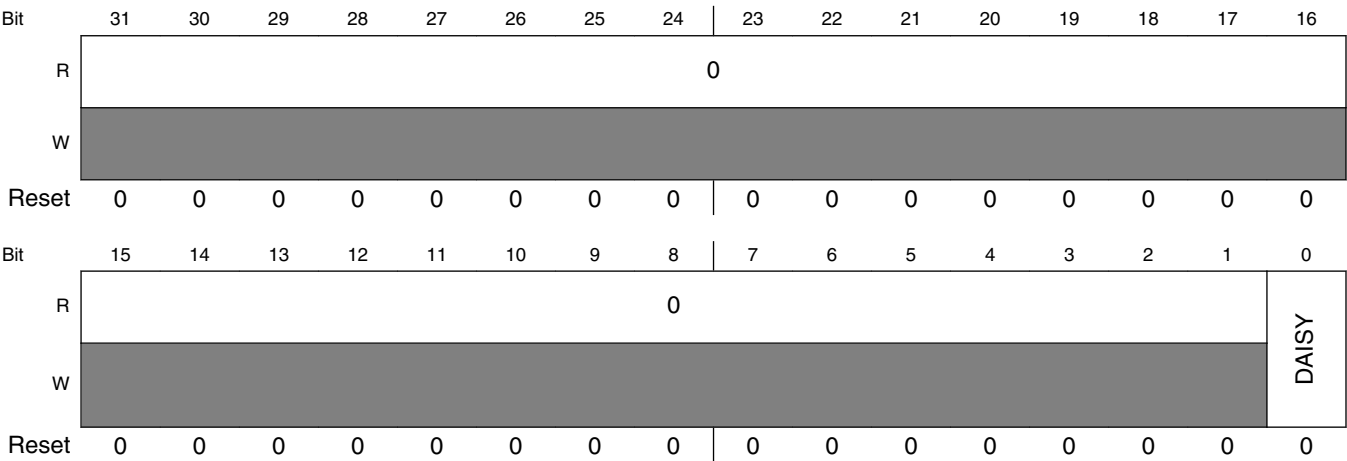
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_CSI\_CSI\_DATA13\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>LCD_DATA20_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT20 for CSI_DATA13. 1 <b>SD3_DATA1_ALT3</b> — Selecting ALT3 mode of pad SD3_DAT1 for CSI_DATA13.

### 30.4.406 Select Input Register (IOMUXC\_CSI\_CSI\_DATA14\_SELECT\_INPUT)

Address: 20E\_0000h base + 668h offset = 20E\_0668h



IOMUXC\_CSI\_CSI\_DATA14\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>LCD_DATA19_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT19 for CSI_DATA14. 1 <b>SD3_DATA2_ALT3</b> — Selecting ALT3 mode of pad SD3_DAT2 for CSI_DATA14.

### 30.4.407 Select Input Register (IOMUXC\_CSI\_CSI\_DATA15\_SELECT\_INPUT)

Address: 20E\_0000h base + 66Ch offset = 20E\_066Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_CSI\_CSI\_DATA15\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>LCD_DATA18_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT18 for CSI_DATA15. 1 <b>SD3_DATA3_ALT3</b> — Selecting ALT3 mode of pad SD3_DAT3 for CSI_DATA15.

### 30.4.408 Select Input Register (IOMUXC\_CSI\_CSI\_HSYNC\_SELECT\_INPUT)

Address: 20E\_0000h base + 670h offset = 20E\_0670h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_CSI\_CSI\_HSYNC\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>ECSPI2_MOSI_ALT3</b> — Selecting ALT3 mode of pad ECSPI2_MOSI for CSI_HSYNC. 01 <b>EPDC_GDOE_ALT3</b> — Selecting ALT3 mode of pad EPDC_GDOE for CSI_HSYNC. 10 <b>LCD_DATA05_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT5 for CSI_HSYNC.

### 30.4.409 Select Input Register (IOMUXC\_CSI\_CSI\_PIXCLK\_SELECT\_INPUT)

Address: 20E\_0000h base + 674h offset = 20E\_0674h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_CSI\_CSI\_PIXCLK\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>ECSPI2_SCLK_ALT3</b> — Selecting ALT3 mode of pad ECSPI2_SCLK for CSI_PIXCLK. 01 <b>EPDC_GDCLK_ALT3</b> — Selecting ALT3 mode of pad EPDC_GDCLK for CSI_PIXCLK. 10 <b>LCD_DATA06_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT6 for CSI_PIXCLK.

### 30.4.410 Select Input Register (IOMUXC\_CSI\_CSI\_VSYNC\_SELECT\_INPUT)

Address: 20E\_0000h base + 678h offset = 20E\_0678h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_CSI\_CSI\_VSYNC\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>ECSPI2_SS0_ALT3</b> — Selecting ALT3 mode of pad ECSPI2_SS0 for CSI_VSYNC. 01 <b>EPDC_GDSP_ALT3</b> — Selecting ALT3 mode of pad EPDC_GDSP for CSI_VSYNC. 10 <b>LCD_DATA04_ALT2</b> — Selecting ALT2 mode of pad LCD_DAT4 for CSI_VSYNC.

### 30.4.411 Select Input Register (IOMUXC\_ECSP11\_CSPI\_CLK\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 67Ch offset = 20E\_067Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



**IOMUXC\_ECSP11\_CSPI\_CLK\_IN\_SELECT\_INPUT field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>ECSP11_SCLK_ALT0</b> — Selecting ALT0 mode of pad ECSP11_SCLK for ECSP11_SCLK. 1 <b>LCD_DATA03_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT3 for ECSP11_SCLK.

### 30.4.412 Select Input Register (IOMUXC\_ECSP11\_DATAREADY\_B\_SELECT\_INPUT)

Address: 20E\_0000h base + 680h offset = 20E\_0680h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_ECSP11\_DATAREADY\_B\_SELECT\_INPUT field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>I2C2_SCL_ALT6</b> — Selecting ALT6 mode of pad I2C2_SCL for ECSP11_RDY. 1 <b>LCD_DATA07_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT7 for ECSP11_RDY.

30.4.413 Select Input Register  
(IOMUXC\_ECSP11\_MISO\_SELECT\_INPUT)

Address: 20E\_0000h base + 684h offset = 20E\_0684h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_ECSP11\_MISO\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>ECSP11_MISO_ALT0</b> — Selecting ALT0 mode of pad ECSP11_MISO for ECSP11_MISO. 1 <b>LCD_DATA01_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT1 for ECSP11_MISO.

30.4.414 Select Input Register  
(IOMUXC\_ECSP11\_MOSI\_SELECT\_INPUT)

Address: 20E\_0000h base + 688h offset = 20E\_0688h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_ECSP11\_MOSI\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>ECSP11_MOSI_ALT0</b> — Selecting ALT0 mode of pad ECSP11_MOSI for ECSP11_MOSI. 1 <b>LCD_DATA00_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT0 for ECSP11_MOSI.

30.4.415 Select Input Register  
(IOMUXC\_ECSP11\_SS0\_SELECT\_INPUT)

Address: 20E\_0000h base + 68Ch offset = 20E\_068Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_ECSP11\_SS0\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>ECSP11_SS0_ALT0</b> — Selecting ALT0 mode of pad ECSP11_SS0 for ECSP11_SS0. 1 <b>LCD_DATA02_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT2 for ECSP11_SS0.

30.4.416 Select Input Register  
(IOMUXC\_ECSP1\_SS1\_SELECT\_INPUT)

Address: 20E\_0000h base + 690h offset = 20E\_0690h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_ECSP1\_SS1\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>I2C1_SCL_ALT6</b> — Selecting ALT6 mode of pad I2C1_SCL for ECSP1_SS1. 1 <b>LCD_DATA04_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT4 for ECSP1_SS1.

30.4.417 Select Input Register  
(IOMUXC\_ECSP1\_SS2\_SELECT\_INPUT)

Address: 20E\_0000h base + 694h offset = 20E\_0694h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_ECSP1\_SS2\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>I2C1_SDA_ALT6</b> — Selecting ALT6 mode of pad I2C1_SDA for ECSP1_SS2. 1 <b>LCD_DATA05_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT5 for ECSP1_SS2.

### 30.4.418 Select Input Register (IOMUXC\_ECSP1\_SS3\_SELECT\_INPUT)

Address: 20E\_0000h base + 698h offset = 20E\_0698h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_ECSP1\_SS3\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>ECSPI2_SS0_ALT1</b> — Selecting ALT1 mode of pad ECSPI2_SS0 for ECSP1_SS3. 1 <b>LCD_DATA06_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT6 for ECSP1_SS3.

### 30.4.419 Select Input Register (IOMUXC\_ECSP12\_CSPI\_CLK\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 69Ch offset = 20E\_069Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_ECSPi2\_CSPI\_CLK\_IN\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>ECSPi2_SCLK_ALT0</b> — Selecting ALT0 mode of pad ECSPi2_SCLK for ECSPi2_SCLK. 01 <b>EPDC_SDSHR_ALT1</b> — Selecting ALT1 mode of pad EPDC_SDSHR for ECSPi2_SCLK. 10 <b>LCD_DATA08_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT8 for ECSPi2_SCLK.

### 30.4.420 Select Input Register (IOMUXC\_ECSPi2\_MISO\_SELECT\_INPUT)

Address: 20E\_0000h base + 6A0h offset = 20E\_06A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_ECSPi2\_MISO\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>ECSPi2_MISO_ALT0</b> — Selecting ALT0 mode of pad ECSPi2_MISO for ECSPi2_MISO. 01 <b>EPDC_SDLE_ALT1</b> — Selecting ALT1 mode of pad EPDC_SDLE for ECSPi2_MISO. 10 <b>LCD_DATA10_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT10 for ECSPi2_MISO.



### 30.4.421 Select Input Register (IOMUXC\_ECSPi2\_MOSI\_SELECT\_INPUT)

Address: 20E\_0000h base + 6A4h offset = 20E\_06A4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_ECSPi2\_MOSI\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>ECSPi2_MOSI_ALT0</b> — Selecting ALT0 mode of pad ECSPi2_MOSI for ECSPi2_MOSI. 01 <b>EPDC_SDCLK_ALT1</b> — Selecting ALT1 mode of pad EPDC_SDCLK for ECSPi2_MOSI. 10 <b>LCD_DATA09_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT9 for ECSPi2_MOSI.

### 30.4.422 Select Input Register (IOMUXC\_ECSPi2\_SS0\_SELECT\_INPUT)

Address: 20E\_0000h base + 6A8h offset = 20E\_06A8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_ECSPi2\_SS0\_SELECT\_INPUT field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>ECSPi2_SS0_ALT0</b> — Selecting ALT0 mode of pad ECSPi2_SS0 for ECSPi2_SS0. 1 <b>EPDC_SDOE_ALT1</b> — Selecting ALT1 mode of pad EPDC_SDOE for ECSPi2_SS0.

### 30.4.423 Select Input Register (IOMUXC\_ECSPi2\_SS1\_SELECT\_INPUT)

Address: 20E\_0000h base + 6ACh offset = 20E\_06ACh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_ECSPi2\_SS1\_SELECT\_INPUT field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_SDCE0_ALT1</b> — Selecting ALT1 mode of pad EPDC_SDCE0 for ECSPi2_SS1. 1 <b>LCD_DATA11_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT11 for ECSPi2_SS1.

### 30.4.424 Select Input Register (IOMUXC\_ECSPi3\_CSPI\_CLK\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 6B0h offset = 20E\_06B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_ECSPi3\_CSPI\_CLK\_IN\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>AUD_TXD_ALT1</b> — Selecting ALT1 mode of pad AUD_TXD for ECSPi3_SCLK. 01 <b>EPDC_DATA11_ALT1</b> — Selecting ALT1 mode of pad EPDC_D11 for ECSPi3_SCLK. 10 <b>SD2_CLK_ALT2</b> — Selecting ALT2 mode of pad SD2_CLK for ECSPi3_SCLK.

### 30.4.425 Select Input Register (IOMUXC\_ECSPi3\_DATAREADY\_B\_SELECT\_INPUT)

Address: 20E\_0000h base + 6B4h offset = 20E\_06B4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_ECSPi3\_DATAREADY\_B\_SELECT\_INPUT field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>AUD_MCLK_ALT2</b> — Selecting ALT2 mode of pad AUD_MCLK for ECSPi3_RDY. 1 <b>EPDC_DATA15_ALT6</b> — Selecting ALT6 mode of pad EPDC_D15 for ECSPi3_RDY.

### 30.4.426 Select Input Register (IOMUXC\_ECSPi3\_MISO\_SELECT\_INPUT)

Address: 20E\_0000h base + 6B8h offset = 20E\_06B8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_ECSPi3\_MISO\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>AUD_TXC_ALT1</b> — Selecting ALT1 mode of pad AUD_TXC for ECSPi3_MISO. 01 <b>EPDC_DATA09_ALT1</b> — Selecting ALT1 mode of pad EPDC_D9 for ECSPi3_MISO. 10 <b>SD2_DATA1_ALT2</b> — Selecting ALT2 mode of pad SD2_DAT1 for ECSPi3_MISO.

### 30.4.427 Select Input Register (IOMUXC\_ECSPi3\_MOSI\_SELECT\_INPUT)

Address: 20E\_0000h base + 6BCh offset = 20E\_06BCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_ECSPi3\_MOSI\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>AUD_RXD_ALT1</b> — Selecting ALT1 mode of pad AUD_RXD for ECSPi3_MOSI. 01 <b>EPDC_DATA08_ALT1</b> — Selecting ALT1 mode of pad EPDC_D8 for ECSPi3_MOSI. 10 <b>SD2_DATA0_ALT2</b> — Selecting ALT2 mode of pad SD2_DAT0 for ECSPi3_MOSI.

### 30.4.428 Select Input Register (IOMUXC\_ECSPi3\_SS0\_SELECT\_INPUT)

Address: 20E\_0000h base + 6C0h offset = 20E\_06C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_ECSPi3\_SS0\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**IOMUXC\_ECSPi3\_SS0\_SELECT\_INPUT field descriptions (continued)**

Field	Description
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>AUD_RXFS_ALT6</b> — Selecting ALT6 mode of pad AUD_RXFS for ECSPi3_SS0. 01 <b>EPDC_DATA10_ALT1</b> — Selecting ALT1 mode of pad EPDC_D10 for ECSPi3_SS0. 10 <b>SD2_CMD_ALT2</b> — Selecting ALT2 mode of pad SD2_CMD for ECSPi3_SS0.

### 30.4.429 Select Input Register (IOMUXC\_ECSPi3\_SS1\_SELECT\_INPUT)

Address: 20E\_0000h base + 6C4h offset = 20E\_06C4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_ECSPi3\_SS1\_SELECT\_INPUT field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>AUD_RXC_ALT6</b> — Selecting ALT6 mode of pad AUD_RXC for ECSPi3_SS1. 1 <b>EPDC_DATA12_ALT6</b> — Selecting ALT6 mode of pad EPDC_D12 for ECSPi3_SS1.

30.4.430 Select Input Register  
(IOMUXC\_ECSPi3\_SS2\_SELECT\_INPUT)

Address: 20E\_0000h base + 6C8h offset = 20E\_06C8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_ECSPi3\_SS2\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_DATA13_ALT6</b> — Selecting ALT6 mode of pad EPDC_D13 for ECSPi3_SS2. 1 <b>I2C1_SCL_ALT2</b> — Selecting ALT2 mode of pad I2C1_SCL for ECSPi3_SS2.

### 30.4.431 Select Input Register (IOMUXC\_ECSPi3\_SS3\_SELECT\_INPUT)

Address: 20E\_0000h base + 6CCh offset = 20E\_06CCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_ECSPi3\_SS3\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_DATA14_ALT6</b> — Selecting ALT6 mode of pad EPDC_D14 for ECSPi3_SS3. 1 <b>I2C1_SDA_ALT2</b> — Selecting ALT2 mode of pad I2C1_SDA for ECSPi3_SS3.

### 30.4.432 Select Input Register (IOMUXC\_ECSPi4\_CSPI\_CLK\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 6D0h offset = 20E\_06D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



**IOMUXC\_ECSPi4\_CSPI\_CLK\_IN\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_DATA03_ALT1</b> — Selecting ALT1 mode of pad EPDC_D3 for ECSPi4_SCLK. 01 <b>FEC_TX_CLK_ALT3</b> — Selecting ALT3 mode of pad FEC_TX_CLK for ECSPi4_SCLK. 10 <b>KEY_ROW2_ALT1</b> — Selecting ALT1 mode of pad KEY_ROW2 for ECSPi4_SCLK.

**30.4.433 Select Input Register (IOMUXC\_ECSPi4\_MISO\_SELECT\_INPUT)**

Address: 20E\_0000h base + 6D4h offset = 20E\_06D4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_ECSPi4\_MISO\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_DATA01_ALT1</b> — Selecting ALT1 mode of pad EPDC_D1 for ECSPi4_MISO. 01 <b>FEC_CRS_DV_ALT3</b> — Selecting ALT3 mode of pad FEC_CRS_DV for ECSPi4_MISO. 10 <b>KEY_ROW1_ALT1</b> — Selecting ALT1 mode of pad KEY_ROW1 for ECSPi4_MISO.

### 30.4.434 Select Input Register (IOMUXC\_ECSPi4\_MOSI\_SELECT\_INPUT)

Address: 20E\_0000h base + 6D8h offset = 20E\_06D8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_ECSPi4\_MOSI\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_DATA00_ALT1</b> — Selecting ALT1 mode of pad EPDC_D0 for ECSPi4_MOSI. 01 <b>FEC_RX_ER_ALT3</b> — Selecting ALT3 mode of pad FEC_RX_ER for ECSPi4_MOSI. 10 <b>KEY_COL1_ALT1</b> — Selecting ALT1 mode of pad KEY_COL1 for ECSPi4_MOSI.

### 30.4.435 Select Input Register (IOMUXC\_ECSPi4\_SS0\_SELECT\_INPUT)

Address: 20E\_0000h base + 6DCh offset = 20E\_06DCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_ECSPi4\_SS0\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**IOMUXC\_ECSPi4\_SS0\_SELECT\_INPUT field descriptions (continued)**

Field	Description
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_DATA02_ALT1</b> — Selecting ALT1 mode of pad EPDC_D2 for ECSPi4_SS0. 01 <b>FEC_MDIO_ALT3</b> — Selecting ALT3 mode of pad FEC_MDIO for ECSPi4_SS0. 10 <b>KEY_COL2_ALT1</b> — Selecting ALT1 mode of pad KEY_COL2 for ECSPi4_SS0.

### 30.4.436 Select Input Register (IOMUXC\_ECSPi4\_SS1\_SELECT\_INPUT)

Address: 20E\_0000h base + 6E0h offset = 20E\_06E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_ECSPi4\_SS1\_SELECT\_INPUT field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_DATA04_ALT1</b> — Selecting ALT1 mode of pad EPDC_D4 for ECSPi4_SS1. 1 <b>FEC_RX_DATA1_ALT3</b> — Selecting ALT3 mode of pad FEC_RXD1 for ECSPi4_SS1.

### 30.4.437 Select Input Register (IOMUXC\_ECSPi4\_SS2\_SELECT\_INPUT)

Address: 20E\_0000h base + 6E4h offset = 20E\_06E4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_ECSPi4\_SS2\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_DATA05_ALT1</b> — Selecting ALT1 mode of pad EPDC_D5 for ECSPi4_SS2. 1 <b>FEC_TX_DATA0_ALT3</b> — Selecting ALT3 mode of pad FEC_TXD0 for ECSPi4_SS2.

30.4.438 Select Input Register  
(IOMUXC\_EPDC\_EPDC\_PWR\_IRQ\_SELECT\_INPUT)

Address: 20E\_0000h base + 6E8h offset = 20E\_06E8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_EPDC\_EPDC\_PWR\_IRQ\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_DATA13_ALT2</b> — Selecting ALT2 mode of pad EPDC_D13 for EPDC_PWR_IRQ. 1 <b>EPDC_PWR_IRQ_ALT0</b> — Selecting ALT0 mode of pad EPDC_PWRINT for EPDC_PWR_IRQ.

### 30.4.439 Select Input Register (IOMUXC\_EPDC\_EPDC\_PWR\_STAT\_SELECT\_INPUT)

Address: 20E\_0000h base + 6ECh offset = 20E\_06ECh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_EPDC\_EPDC\_PWR\_STAT\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_DATA14_ALT2</b> — Selecting ALT2 mode of pad EPDC_D14 for EPDC_PWR_STAT. 1 <b>EPDC_PWR_STAT_ALT0</b> — Selecting ALT0 mode of pad EPDC_PWRSTAT for EPDC_PWR_STAT.

### 30.4.440 Select Input Register (IOMUXC\_FEC\_FEC\_COL\_SELECT\_INPUT)

Address: 20E\_0000h base + 6F0h offset = 20E\_06F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_FEC\_FEC\_COL\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>FEC_RX_DATA1_ALT6</b> — Selecting ALT6 mode of pad FEC_RXD1 for FEC_COL. 01 <b>SD2_DATA2_ALT2</b> — Selecting ALT2 mode of pad SD2_DAT2 for FEC_COL. 10 <b>UART1_RXD_ALT3</b> — Selecting ALT3 mode of pad UART1_RXD for FEC_COL.

### 30.4.441 Select Input Register (IOMUXC\_FEC\_FEC\_MDI\_SELECT\_INPUT)

Address: 20E\_0000h base + 6F4h offset = 20E\_06F4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_FEC\_FEC\_MDI\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>AUD_RXFS_ALT3</b> — Selecting ALT3 mode of pad AUD_RXFS for FEC_MDIO. 01 <b>FEC_MDIO_ALT0</b> — Selecting ALT0 mode of pad FEC_MDIO for FEC_MDIO. 10 <b>SD1_CLK_ALT1</b> — Selecting ALT1 mode of pad SD1_CLK for FEC_MDIO.

### 30.4.442 Select Input Register (IOMUXC\_FEC\_FEC\_RX\_DATA0\_SELECT\_INPUT)

Address: 20E\_0000h base + 6F8h offset = 20E\_06F8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_FEC\_FEC\_RX\_DATA0\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>FEC_RX_DATA0_ALT0</b> — Selecting ALT0 mode of pad FEC_RXD0 for FEC_RX_DATA0. 01 <b>I2C1_SCL_ALT3</b> — Selecting ALT3 mode of pad I2C1_SCL for FEC_RX_DATA0. 10 <b>SD1_DATA5_ALT1</b> — Selecting ALT1 mode of pad SD1_DAT5 for FEC_RX_DATA0.

### 30.4.443 Select Input Register (IOMUXC\_FEC\_FEC\_RX\_DATA1\_SELECT\_INPUT)

Address: 20E\_0000h base + 6FCh offset = 20E\_06FCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_FEC\_FEC\_RX\_DATA1\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...



**IOMUXC\_FEC\_FEC\_RX\_DATA1\_SELECT\_INPUT field descriptions (continued)**

Field	Description
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>AUD_TXFS_ALT3</b> — Selecting ALT3 mode of pad AUD_TXFS for FEC_RX_DATA1. 01 <b>FEC_RX_DATA1_ALT0</b> — Selecting ALT0 mode of pad FEC_RXD1 for FEC_RX_DATA1. 10 <b>SD1_DATA2_ALT1</b> — Selecting ALT1 mode of pad SD1_DAT2 for FEC_RX_DATA1.

### 30.4.444 Select Input Register (IOMUXC\_FEC\_FEC\_RX\_CLK\_SELECT\_INPUT)

Address: 20E\_0000h base + 700h offset = 20E\_0700h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_FEC\_FEC\_RX\_CLK\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>FEC_TX_DATA1_ALT6</b> — Selecting ALT6 mode of pad FEC_TXD1 for FEC_RX_CLK. 01 <b>SD2_DATA3_ALT2</b> — Selecting ALT2 mode of pad SD2_DAT3 for FEC_RX_CLK. 10 <b>UART1_TXD_ALT3</b> — Selecting ALT3 mode of pad UART1_TXD for FEC_RX_CLK.

### 30.4.445 Select Input Register (IOMUXC\_FEC\_FEC\_RX\_DV\_SELECT\_INPUT)

Address: 20E\_0000h base + 704h offset = 20E\_0704h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_FEC\_FEC\_RX\_DV\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>AUD_TXC_ALT3</b> — Selecting ALT3 mode of pad AUD_TXC for FEC_RX_DV. 01 <b>FEC_CRS_DV_ALT0</b> — Selecting ALT0 mode of pad FEC_CRS_DV for FEC_RX_DV. 10 <b>SD1_DATA1_ALT1</b> — Selecting ALT1 mode of pad SD1_DAT1 for FEC_RX_DV.

### 30.4.446 Select Input Register (IOMUXC\_FEC\_FEC\_RX\_ER\_SELECT\_INPUT)

Address: 20E\_0000h base + 708h offset = 20E\_0708h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_FEC\_FEC\_RX\_ER\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**IOMUXC\_FEC\_FEC\_RX\_ER\_SELECT\_INPUT field descriptions (continued)**

Field	Description
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>AUD_RXD_ALT3</b> — Selecting ALT3 mode of pad AUD_RXD for FEC_RX_ER. 01 <b>FEC_RX_ER_ALT0</b> — Selecting ALT0 mode of pad FEC_RX_ER for FEC_RX_ER. 10 <b>SD1_DATA0_ALT1</b> — Selecting ALT1 mode of pad SD1_DATA0 for FEC_RX_ER.

### 30.4.447 Select Input Register (IOMUXC\_FEC\_FEC\_TX\_CLK\_SELECT\_INPUT)

Address: 20E\_0000h base + 70Ch offset = 20E\_070Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_FEC\_FEC\_TX\_CLK\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>AUD_RXC_ALT3</b> — Selecting ALT3 mode of pad AUD_RXC for FEC_TX_CLK. 01 <b>FEC_TX_CLK_ALT0</b> — Selecting ALT0 mode of pad FEC_TX_CLK for FEC_TX_CLK. 10 <b>SD1_CMD_ALT1</b> — Selecting ALT1 mode of pad SD1_CMD for FEC_TX_CLK.

30.4.448 Select Input Register  
(IOMUXC\_GPT\_CAPIN1\_SELECT\_INPUT)

Address: 20E\_0000h base + 710h offset = 20E\_0710h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_GPT\_CAPIN1\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>FEC_MDIO_ALT4</b> — Selecting ALT4 mode of pad FEC_MDIO for GPT_CAPTURE1. 1 <b>LCD_DATA18_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT18 for GPT_CAPTURE1.

30.4.449 Select Input Register  
(IOMUXC\_GPT\_CAPIN2\_SELECT\_INPUT)

Address: 20E\_0000h base + 714h offset = 20E\_0714h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_GPT\_CAPIN2\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>FEC_TX_CLK_ALT4</b> — Selecting ALT4 mode of pad FEC_TX_CLK for GPT_CAPTURE2. 1 <b>LCD_DATA19_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT19 for GPT_CAPTURE2.

### 30.4.450 Select Input Register (IOMUXC\_GPT\_CLKIN\_SELECT\_INPUT)

Address: 20E\_0000h base + 718h offset = 20E\_0718h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_GPT\_CLKIN\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>FEC_TX_DATA0_ALT4</b> — Selecting ALT4 mode of pad FEC_TXD0 for GPT_CLKIN. 1 <b>LCD_DATA23_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT23 for GPT_CLKIN.

### 30.4.451 Select Input Register (IOMUXC\_I2C1\_SCL\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 71Ch offset = 20E\_071Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_I2C1\_SCL\_IN\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>AUD_RXFS_ALT1</b> — Selecting ALT1 mode of pad AUD_RXFS for I2C1_SCL. 01 <b>USB_H_DATA_ALT1</b> — Selecting ALT1 mode of pad HSIC_DAT for I2C1_SCL. 10 <b>I2C1_SCL_ALT0</b> — Selecting ALT0 mode of pad I2C1_SCL for I2C1_SCL.

### 30.4.452 Select Input Register (IOMUXC\_I2C1\_SDA\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 720h offset = 20E\_0720h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_I2C1\_SDA\_IN\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>AUD_RXC_ALT1</b> — Selecting ALT1 mode of pad AUD_RXC for I2C1_SDA. 01 <b>USB_H_STROBE_ALT1</b> — Selecting ALT1 mode of pad HSIC_STROBE for I2C1_SDA. 10 <b>I2C1_SDA_ALT0</b> — Selecting ALT0 mode of pad I2C1_SDA for I2C1_SDA.

### 30.4.453 Select Input Register (IOMUXC\_I2C2\_SCL\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 724h offset = 20E\_0724h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_I2C2\_SCL\_IN\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_SDCLK_ALT2</b> — Selecting ALT2 mode of pad EPDC_SDCLK for I2C2_SCL. 01 <b>I2C2_SCL_ALT0</b> — Selecting ALT0 mode of pad I2C2_SCL for I2C2_SCL. 10 <b>KEY_COLO_ALT1</b> — Selecting ALT1 mode of pad KEY_COLO for I2C2_SCL. 11 <b>LCD_DATA16_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT16 for I2C2_SCL.

### 30.4.454 Select Input Register (IOMUXC\_I2C2\_SDA\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 728h offset = 20E\_0728h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



**IOMUXC\_I2C2\_SDA\_IN\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_SDLE_ALT2</b> — Selecting ALT2 mode of pad EPDC_SDLE for I2C2_SDA. 01 <b>I2C2_SDA_ALT0</b> — Selecting ALT0 mode of pad I2C2_SDA for I2C2_SDA. 10 <b>KEY_ROW0_ALT1</b> — Selecting ALT1 mode of pad KEY_ROW0 for I2C2_SDA. 11 <b>LCD_DATA17_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT17 for I2C2_SDA.

### 30.4.455 Select Input Register (IOMUXC\_I2C3\_SCL\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 72Ch offset = 20E\_072Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_I2C3\_SCL\_IN\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>AUD_RXFS_ALT4</b> — Selecting ALT4 mode of pad AUD_RXFS for I2C3_SCL. 01 <b>EPDC_SDCE2_ALT1</b> — Selecting ALT1 mode of pad EPDC_SDCE2 for I2C3_SCL. 10 <b>REF_CLK_24M_ALT1</b> — Selecting ALT1 mode of pad REF_CLK_24M for I2C3_SCL.

### 30.4.456 Select Input Register (IOMUXC\_I2C3\_SDA\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 730h offset = 20E\_0730h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_I2C3\_SDA\_IN\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>AUD_RXC_ALT4</b> — Selecting ALT4 mode of pad AUD_RXC for I2C3_SDA. 01 <b>EPDC_SDCE3_ALT1</b> — Selecting ALT1 mode of pad EPDC_SDCE3 for I2C3_SDA. 10 <b>REF_CLK_32K_ALT1</b> — Selecting ALT1 mode of pad REF_CLK_32K for I2C3_SDA.

### 30.4.457 Select Input Register (IOMUXC\_KEY\_COL0\_SELECT\_INPUT)

Address: 20E\_0000h base + 734h offset = 20E\_0734h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_KEY\_COL0\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**IOMUXC\_KEY\_COL0\_SELECT\_INPUT field descriptions (continued)**

Field	Description
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>KEY_COL0_ALT0</b> — Selecting ALT0 mode of pad KEY_COL0 for KEY_COL0. 01 <b>LCD_DATA08_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT8 for KEY_COL0. 10 <b>SD1_CLK_ALT2</b> — Selecting ALT2 mode of pad SD1_CLK for KEY_COL0.

### 30.4.458 Select Input Register (IOMUXC\_KEY\_COL1\_SELECT\_INPUT)

Address: 20E\_0000h base + 738h offset = 20E\_0738h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_KEY\_COL1\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>KEY_COL1_ALT0</b> — Selecting ALT0 mode of pad KEY_COL1 for KEY_COL1. 01 <b>LCD_DATA10_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT10 for KEY_COL1. 10 <b>SD1_DATA0_ALT2</b> — Selecting ALT2 mode of pad SD1_DAT0 for KEY_COL1.

### 30.4.459 Select Input Register (IOMUXC\_KEY\_COL2\_SELECT\_INPUT)

Address: 20E\_0000h base + 73Ch offset = 20E\_073Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_KEY\_COL2\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>KEY_COL2_ALT0</b> — Selecting ALT0 mode of pad KEY_COL2 for KEY_COL2. 01 <b>LCD_DATA12_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT12 for KEY_COL2. 10 <b>SD1_DATA2_ALT2</b> — Selecting ALT2 mode of pad SD1_DAT2 for KEY_COL2.

### 30.4.460 Select Input Register (IOMUXC\_KEY\_COL3\_SELECT\_INPUT)

Address: 20E\_0000h base + 740h offset = 20E\_0740h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_KEY\_COL3\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**IOMUXC\_KEY\_COL3\_SELECT\_INPUT field descriptions (continued)**

Field	Description
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>KEY_COL3_ALT0</b> — Selecting ALT0 mode of pad KEY_COL3 for KEY_COL3. 01 <b>LCD_DATA14_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT14 for KEY_COL3. 10 <b>SD1_DATA4_ALT2</b> — Selecting ALT2 mode of pad SD1_DAT4 for KEY_COL3.

### 30.4.461 Select Input Register (IOMUXC\_KEY\_COL4\_SELECT\_INPUT)

Address: 20E\_0000h base + 744h offset = 20E\_0744h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_KEY\_COL4\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>KEY_COL4_ALT0</b> — Selecting ALT0 mode of pad KEY_COL4 for KEY_COL4. 01 <b>LCD_DATA16_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT16 for KEY_COL4. 10 <b>SD1_DATA6_ALT2</b> — Selecting ALT2 mode of pad SD1_DAT6 for KEY_COL4.

### 30.4.462 Select Input Register (IOMUXC\_KEY\_COL5\_SELECT\_INPUT)

Address: 20E\_0000h base + 748h offset = 20E\_0748h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_KEY\_COL5\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>KEY_COL5_ALT0</b> — Selecting ALT0 mode of pad KEY_COL5 for KEY_COL5. 01 <b>LCD_DATA18_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT18 for KEY_COL5. 10 <b>SD3_CLK_ALT2</b> — Selecting ALT2 mode of pad SD3_CLK for KEY_COL5.

### 30.4.463 Select Input Register (IOMUXC\_KEY\_COL6\_SELECT\_INPUT)

Address: 20E\_0000h base + 74Ch offset = 20E\_074Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_KEY\_COL6\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**IOMUXC\_KEY\_COL6\_SELECT\_INPUT field descriptions (continued)**

Field	Description
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>KEY_COL6_ALT0</b> — Selecting ALT0 mode of pad KEY_COL6 for KEY_COL6. 01 <b>LCD_DATA20_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT20 for KEY_COL6. 10 <b>SD3_DATA0_ALT2</b> — Selecting ALT2 mode of pad SD3_DAT0 for KEY_COL6.

### 30.4.464 Select Input Register (IOMUXC\_KEY\_COL7\_SELECT\_INPUT)

Address: 20E\_0000h base + 750h offset = 20E\_0750h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_KEY\_COL7\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>KEY_COL7_ALT0</b> — Selecting ALT0 mode of pad KEY_COL7 for KEY_COL7. 01 <b>LCD_DATA22_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT22 for KEY_COL7. 10 <b>SD3_DATA2_ALT2</b> — Selecting ALT2 mode of pad SD3_DAT2 for KEY_COL7.

### 30.4.465 Select Input Register (IOMUXC\_KEY\_ROW0\_SELECT\_INPUT)

Address: 20E\_0000h base + 754h offset = 20E\_0754h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_KEY\_ROW0\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>KEY_ROW0_ALT0</b> — Selecting ALT0 mode of pad KEY_ROW0 for KEY_ROW0. 01 <b>LCD_DATA09_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT9 for KEY_ROW0. 10 <b>SD1_CMD_ALT2</b> — Selecting ALT2 mode of pad SD1_CMD for KEY_ROW0.

### 30.4.466 Select Input Register (IOMUXC\_KEY\_ROW1\_SELECT\_INPUT)

Address: 20E\_0000h base + 758h offset = 20E\_0758h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_KEY\_ROW1\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...



**IOMUXC\_KEY\_ROW1\_SELECT\_INPUT field descriptions (continued)**

Field	Description
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>KEY_ROW1_ALT0</b> — Selecting ALT0 mode of pad KEY_ROW1 for KEY_ROW1. 01 <b>LCD_DATA11_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT11 for KEY_ROW1. 10 <b>SD1_DATA1_ALT2</b> — Selecting ALT2 mode of pad SD1_DAT1 for KEY_ROW1.

### 30.4.467 Select Input Register (IOMUXC\_KEY\_ROW2\_SELECT\_INPUT)

Address: 20E\_0000h base + 75Ch offset = 20E\_075Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_KEY\_ROW2\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>KEY_ROW2_ALT0</b> — Selecting ALT0 mode of pad KEY_ROW2 for KEY_ROW2. 01 <b>LCD_DATA13_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT13 for KEY_ROW2. 10 <b>SD1_DATA3_ALT2</b> — Selecting ALT2 mode of pad SD1_DAT3 for KEY_ROW2.

### 30.4.468 Select Input Register (IOMUXC\_KEY\_ROW3\_SELECT\_INPUT)

Address: 20E\_0000h base + 760h offset = 20E\_0760h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_KEY\_ROW3\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>KEY_ROW3_ALT0</b> — Selecting ALT0 mode of pad KEY_ROW3 for KEY_ROW3. 01 <b>LCD_DATA15_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT15 for KEY_ROW3. 10 <b>SD1_DATA5_ALT2</b> — Selecting ALT2 mode of pad SD1_DAT5 for KEY_ROW3.

### 30.4.469 Select Input Register (IOMUXC\_KEY\_ROW4\_SELECT\_INPUT)

Address: 20E\_0000h base + 764h offset = 20E\_0764h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_KEY\_ROW4\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**IOMUXC\_KEY\_ROW4\_SELECT\_INPUT field descriptions (continued)**

Field	Description
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>KEY_ROW4_ALT0</b> — Selecting ALT0 mode of pad KEY_ROW4 for KEY_ROW4. 01 <b>LCD_DATA17_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT17 for KEY_ROW4. 10 <b>SD1_DATA7_ALT2</b> — Selecting ALT2 mode of pad SD1_DAT7 for KEY_ROW4.

### 30.4.470 Select Input Register (IOMUXC\_KEY\_ROW5\_SELECT\_INPUT)

Address: 20E\_0000h base + 768h offset = 20E\_0768h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_KEY\_ROW5\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>KEY_ROW5_ALT0</b> — Selecting ALT0 mode of pad KEY_ROW5 for KEY_ROW5. 01 <b>LCD_DATA19_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT19 for KEY_ROW5. 10 <b>SD3_CMD_ALT2</b> — Selecting ALT2 mode of pad SD3_CMD for KEY_ROW5.

### 30.4.471 Select Input Register (IOMUXC\_KEY\_ROW6\_SELECT\_INPUT)

Address: 20E\_0000h base + 76Ch offset = 20E\_076Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_KEY\_ROW6\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>KEY_ROW6_ALT0</b> — Selecting ALT0 mode of pad KEY_ROW6 for KEY_ROW6. 01 <b>LCD_DATA21_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT21 for KEY_ROW6. 10 <b>SD3_DATA1_ALT2</b> — Selecting ALT2 mode of pad SD3_DAT1 for KEY_ROW6.

### 30.4.472 Select Input Register (IOMUXC\_KEY\_ROW7\_SELECT\_INPUT)

Address: 20E\_0000h base + 770h offset = 20E\_0770h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_KEY\_ROW7\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**IOMUXC\_KEY\_ROW7\_SELECT\_INPUT field descriptions (continued)**

Field	Description
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>KEY_ROW7_ALT0</b> — Selecting ALT0 mode of pad KEY_ROW7 for KEY_ROW7. 01 <b>LCD_DATA23_ALT1</b> — Selecting ALT1 mode of pad LCD_DAT23 for KEY_ROW7. 10 <b>SD3_DATA3_ALT2</b> — Selecting ALT2 mode of pad SD3_DAT3 for KEY_ROW7.

### 30.4.473 Select Input Register (IOMUXC\_LCD\_BUSY\_SELECT\_INPUT)

Address: 20E\_0000h base + 774h offset = 20E\_0774h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_LCD\_BUSY\_SELECT\_INPUT field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>LCD_HSYNC_ALT0</b> — Selecting ALT0 mode of pad LCD_HSYNC for LCD_HSYNC. 1 <b>LCD_RESET_ALT2</b> — Selecting ALT2 mode of pad LCD_RESET for LCD_BUSY.

### 30.4.474 Select Input Register (IOMUXC\_LCD\_DATA00\_SELECT\_INPUT)

Address: 20E\_0000h base + 778h offset = 20E\_0778h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA00\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_COL0_ALT2</b> — Selecting ALT2 mode of pad KEY_COL0 for LCD_DATA00. 1 <b>LCD_DATA00_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT0 for LCD_DATA00.

30.4.475 Select Input Register  
(IOMUXC\_LCD\_DATA01\_SELECT\_INPUT)

Address: 20E\_0000h base + 77Ch offset = 20E\_077Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA01\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_ROW0_ALT2</b> — Selecting ALT2 mode of pad KEY_ROW0 for LCD_DATA01. 1 <b>LCD_DATA01_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT1 for LCD_DATA01.

30.4.476 Select Input Register  
(IOMUXC\_LCD\_DATA02\_SELECT\_INPUT)

Address: 20E\_0000h base + 780h offset = 20E\_0780h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA02\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_COL1_ALT2</b> — Selecting ALT2 mode of pad KEY_COL1 for LCD_DATA02. 1 <b>LCD_DATA02_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT2 for LCD_DATA02.



30.4.477 Select Input Register  
(IOMUXC\_LCD\_DATA03\_SELECT\_INPUT)

Address: 20E\_0000h base + 784h offset = 20E\_0784h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA03\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_ROW1_ALT2</b> — Selecting ALT2 mode of pad KEY_ROW1 for LCD_DATA03. 1 <b>LCD_DATA03_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT3 for LCD_DATA03.

### 30.4.478 Select Input Register (IOMUXC\_LCD\_DATA04\_SELECT\_INPUT)

Address: 20E\_0000h base + 788h offset = 20E\_0788h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA04\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_COL2_ALT2</b> — Selecting ALT2 mode of pad KEY_COL2 for LCD_DATA04. 1 <b>LCD_DATA04_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT4 for LCD_DATA04.

### 30.4.479 Select Input Register (IOMUXC\_LCD\_DATA05\_SELECT\_INPUT)

Address: 20E\_0000h base + 78Ch offset = 20E\_078Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA05\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_ROW2_ALT2</b> — Selecting ALT2 mode of pad KEY_ROW2 for LCD_DATA05. 1 <b>LCD_DATA05_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT5 for LCD_DATA05.

30.4.480 Select Input Register  
(IOMUXC\_LCD\_DATA06\_SELECT\_INPUT)

Address: 20E\_0000h base + 790h offset = 20E\_0790h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA06\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_COL3_ALT2</b> — Selecting ALT2 mode of pad KEY_COL3 for LCD_DATA06. 1 <b>LCD_DATA06_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT6 for LCD_DATA06.

### 30.4.481 Select Input Register (IOMUXC\_LCD\_DATA07\_SELECT\_INPUT)

Address: 20E\_0000h base + 794h offset = 20E\_0794h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA07\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_ROW3_ALT2</b> — Selecting ALT2 mode of pad KEY_ROW3 for LCD_DATA07. 1 <b>LCD_DATA07_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT7 for LCD_DATA07.

30.4.482 Select Input Register  
(IOMUXC\_LCD\_DATA08\_SELECT\_INPUT)

Address: 20E\_0000h base + 798h offset = 20E\_0798h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA08\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_COL4_ALT2</b> — Selecting ALT2 mode of pad KEY_COL4 for LCD_DATA08. 1 <b>LCD_DATA08_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT8 for LCD_DATA08.

30.4.483 Select Input Register  
(IOMUXC\_LCD\_DATA09\_SELECT\_INPUT)

Address: 20E\_0000h base + 79Ch offset = 20E\_079Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA09\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_ROW4_ALT2</b> — Selecting ALT2 mode of pad KEY_ROW4 for LCD_DATA09. 1 <b>LCD_DATA09_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT9 for LCD_DATA09.

30.4.484 Select Input Register  
(IOMUXC\_LCD\_DATA10\_SELECT\_INPUT)

Address: 20E\_0000h base + 7A0h offset = 20E\_07A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA10\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_COL5_ALT2</b> — Selecting ALT2 mode of pad KEY_COL5 for LCD_DATA10. 1 <b>LCD_DATA10_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT10 for LCD_DATA10.



30.4.485 Select Input Register  
(IOMUXC\_LCD\_DATA11\_SELECT\_INPUT)

Address: 20E\_0000h base + 7A4h offset = 20E\_07A4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA11\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_ROW5_ALT2</b> — Selecting ALT2 mode of pad KEY_ROW5 for LCD_DATA11. 1 <b>LCD_DATA11_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT11 for LCD_DATA11.

### 30.4.486 Select Input Register (IOMUXC\_LCD\_DATA12\_SELECT\_INPUT)

Address: 20E\_0000h base + 7A8h offset = 20E\_07A8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA12\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_COL6_ALT2</b> — Selecting ALT2 mode of pad KEY_COL6 for LCD_DATA12. 1 <b>LCD_DATA12_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT12 for LCD_DATA12.

30.4.487 Select Input Register  
(IOMUXC\_LCD\_DATA13\_SELECT\_INPUT)

Address: 20E\_0000h base + 7ACh offset = 20E\_07ACh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA13\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_ROW6_ALT2</b> — Selecting ALT2 mode of pad KEY_ROW6 for LCD_DATA13. 1 <b>LCD_DATA13_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT13 for LCD_DATA13.

30.4.488 Select Input Register  
(IOMUXC\_LCD\_DATA14\_SELECT\_INPUT)

Address: 20E\_0000h base + 7B0h offset = 20E\_07B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA14\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_COL7_ALT2</b> — Selecting ALT2 mode of pad KEY_COL7 for LCD_DATA14. 1 <b>LCD_DATA14_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT14 for LCD_DATA14.

30.4.489 Select Input Register  
(IOMUXC\_LCD\_DATA15\_SELECT\_INPUT)

Address: 20E\_0000h base + 7B4h offset = 20E\_07B4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA15\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_ROW7_ALT2</b> — Selecting ALT2 mode of pad KEY_ROW7 for LCD_DATA15. 1 <b>LCD_DATA15_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT15 for LCD_DATA15.

30.4.490 Select Input Register  
(IOMUXC\_LCD\_DATA16\_SELECT\_INPUT)

Address: 20E\_0000h base + 7B8h offset = 20E\_07B8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA16\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_PWR_CTRL0_ALT2</b> — Selecting ALT2 mode of pad EPDC_PWRCTRL0 for LCD_DATA16. 1 <b>LCD_DATA16_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT16 for LCD_DATA16.

### 30.4.491 Select Input Register (IOMUXC\_LCD\_DATA17\_SELECT\_INPUT)

Address: 20E\_0000h base + 7BCh offset = 20E\_07BCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA17\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_PWR_CTRL1_ALT2</b> — Selecting ALT2 mode of pad EPDC_PWRCTRL1 for LCD_DATA17. 1 <b>LCD_DATA17_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT17 for LCD_DATA17.

30.4.492 Select Input Register  
(IOMUXC\_LCD\_DATA18\_SELECT\_INPUT)

Address: 20E\_0000h base + 7C0h offset = 20E\_07C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA18\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_PWR_CTRL2_ALT2</b> — Selecting ALT2 mode of pad EPDC_PWRCTRL2 for LCD_DATA18. 1 <b>LCD_DATA18_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT18 for LCD_DATA18.



30.4.493 Select Input Register  
(IOMUXC\_LCD\_DATA19\_SELECT\_INPUT)

Address: 20E\_0000h base + 7C4h offset = 20E\_07C4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA19\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_PWR_CTRL3_ALT2</b> — Selecting ALT2 mode of pad EPDC_PWRCTRL3 for LCD_DATA19. 1 <b>LCD_DATA19_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT19 for LCD_DATA19.

### 30.4.494 Select Input Register (IOMUXC\_LCD\_DATA20\_SELECT\_INPUT)

Address: 20E\_0000h base + 7C8h offset = 20E\_07C8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA20\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_PWR_COM_ALT2</b> — Selecting ALT2 mode of pad EPDC_PWRCOM for LCD_DATA20. 1 <b>LCD_DATA20_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT20 for LCD_DATA20.

### 30.4.495 Select Input Register (IOMUXC\_LCD\_DATA21\_SELECT\_INPUT)

Address: 20E\_0000h base + 7CCh offset = 20E\_07CCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA21\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_PWR_IRQ_ALT2</b> — Selecting ALT2 mode of pad EPDC_PWRINT for LCD_DATA21. 1 <b>LCD_DATA21_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT21 for LCD_DATA21.

30.4.496 Select Input Register  
(IOMUXC\_LCD\_DATA22\_SELECT\_INPUT)

Address: 20E\_0000h base + 7D0h offset = 20E\_07D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_LCD\_DATA22\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_PWR_STAT_ALT2</b> — Selecting ALT2 mode of pad EPDC_PWRSTAT for LCD_DATA22. 1 <b>LCD_DATA22_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT22 for LCD_DATA22.

### 30.4.497 Select Input Register (IOMUXC\_LCD\_DATA23\_SELECT\_INPUT)

Address: 20E\_0000h base + 7D4h offset = 20E\_07D4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_LCD\_DATA23\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_PWR_WAKE_ALT2</b> — Selecting ALT2 mode of pad EPDC_PWRWAKEUP for LCD_DATA23. 1 <b>LCD_DATA23_ALT0</b> — Selecting ALT0 mode of pad LCD_DAT23 for LCD_DATA23.

### 30.4.498 Select Input Register (IOMUXC\_SPDIF\_SPDIF\_IN1\_SELECT\_INPUT)

Address: 20E\_0000h base + 7F0h offset = 20E\_07F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SPDIF\_SPDIF\_IN1\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>FEC_TX_EN_ALT2</b> — Selecting ALT2 mode of pad FEC_TX_EN for SPDIF_IN. 01 <b>I2C2_SCL_ALT2</b> — Selecting ALT2 mode of pad I2C2_SCL for SPDIF_IN. 10 <b>SD2_DATA5_ALT4</b> — Selecting ALT4 mode of pad SD2_DAT5 for SPDIF_IN.

### 30.4.499 Select Input Register (IOMUXC\_SPDIF\_TX\_CLK2\_SELECT\_INPUT)

Address: 20E\_0000h base + 7F4h offset = 20E\_07F4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_SPDIF\_TX\_CLK2\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>AUD_MCLK_ALT6</b> — Selecting ALT6 mode of pad AUD_MCLK for SPDIF_EXT_CLK. 01 <b>ECSPi2_SCLK_ALT1</b> — Selecting ALT1 mode of pad ECSPi2_SCLK for SPDIF_EXT_CLK. 10 <b>FEC_REF_CLK_ALT6</b> — Selecting ALT6 mode of pad FEC_REF_CLK for SPDIF_EXT_CLK.

### 30.4.500 Select Input Register (IOMUXC\_UART1\_UART\_RTS\_B\_SELECT\_INPUT)

Address: 20E\_0000h base + 7F8h offset = 20E\_07F8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_UART1\_UART\_RTS\_B\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain. Route RTS_B signal to pad when UART is in DCE mode. Route CTS_B signal to pad when UART is in DTE mode.  0 <b>I2C1_SCL_ALT1</b> — Selecting ALT1 mode of pad I2C1_SCL for UART1_RTS_B. 1 <b>I2C1_SDA_ALT1</b> — Selecting ALT1 mode of pad I2C1_SDA for UART1_CTS_B.

### 30.4.501 Select Input Register (IOMUXC\_UART1\_UART\_RX\_DATA\_SELECT\_INPUT)

Address: 20E\_0000h base + 7FCh offset = 20E\_07FCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_UART1\_UART\_RX\_DATA\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	<p>Input Select (DAISY) Field</p> <p>Selecting Pads Involved in Daisy Chain.</p> <p>Route RX_DATA signal to pad when UART is in DCE mode. Route TX_DATA signal to pad when UART is in DTE mode.</p> <p>0 <b>UART1_RXD_ALT0</b> — Selecting ALT0 mode of pad UART1_RXD for UART1_RX_DATA.</p> <p>1 <b>UART1_TXD_ALT0</b> — Selecting ALT0 mode of pad UART1_TXD for UART1_TX_DATA.</p>

### 30.4.502 Select Input Register (IOMUXC\_UART2\_UART\_RTS\_B\_SELECT\_INPUT)

Address: 20E\_0000h base + 800h offset = 20E\_0800h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																											DAISY				
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	



**IOMUXC\_UART2\_UART\_RTS\_B\_SELECT\_INPUT field descriptions**

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value 0.
2–0 DAISY	<p>Input Select (DAISY) Field</p> <p>Selecting Pads Involved in Daisy Chain.</p> <p>Route RTS_B signal to pad when UART is in DCE mode. Route CTS_B signal to pad when UART is in DTE mode.</p> <p>000 <b>EPDC_DATA14_ALT1</b> — Selecting ALT1 mode of pad EPDC_D14 for UART2_RTS_B.  001 <b>EPDC_DATA15_ALT1</b> — Selecting ALT1 mode of pad EPDC_D15 for UART2_CTS_B.  010 <b>LCD_RESET_ALT4</b> — Selecting ALT4 mode of pad LCD_RESET for UART2_CTS_B.  011 <b>LCD_VSYNC_ALT4</b> — Selecting ALT4 mode of pad LCD_VSYNC for UART2_RTS_B.  100 <b>SD2_DATA6_ALT2</b> — Selecting ALT2 mode of pad SD2_DAT6 for UART2_RTS_B.  101 <b>SD2_DATA7_ALT2</b> — Selecting ALT2 mode of pad SD2_DAT7 for UART2_CTS_B.</p>

### 30.4.503 Select Input Register (IOMUXC\_UART2\_UART\_RX\_DATA\_SELECT\_INPUT)

Address: 20E\_0000h base + 804h offset = 20E\_0804h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																	0															
W																																DAISY
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_UART2\_UART\_RX\_DATA\_SELECT\_INPUT field descriptions**

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value 0.
2–0 DAISY	<p>Input Select (DAISY) Field</p> <p>Selecting Pads Involved in Daisy Chain.</p> <p>Route RX_DATA signal to pad when UART is in DCE mode. Route TX_DATA signal to pad when UART is in DTE mode.</p> <p>000 <b>EPDC_DATA12_ALT1</b> — Selecting ALT1 mode of pad EPDC_D12 for UART2_RX_DATA.  001 <b>EPDC_DATA13_ALT1</b> — Selecting ALT1 mode of pad EPDC_D13 for UART2_TX_DATA.  010 <b>LCD_ENABLE_ALT4</b> — Selecting ALT4 mode of pad LCD_ENABLE for UART2_RX_DATA.  011 <b>LCD_HSYNC_ALT4</b> — Selecting ALT4 mode of pad LCD_HSYNC for UART2_TX_DATA.  100 <b>SD2_DATA4_ALT2</b> — Selecting ALT2 mode of pad SD2_DAT4 for UART2_RX_DATA.  101 <b>SD2_DATA5_ALT2</b> — Selecting ALT2 mode of pad SD2_DAT5 for UART2_TX_DATA.</p>

### 30.4.504 Select Input Register (IOMUXC\_UART3\_UART\_RTS\_B\_SELECT\_INPUT)

Address: 20E\_0000h base + 808h offset = 20E\_0808h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_UART3\_UART\_RTS\_B\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	<p>Input Select (DAISY) Field</p> <p>Selecting Pads Involved in Daisy Chain.</p> <p>Route RTS_B signal to pad when UART is in DCE mode. Route CTS_B signal to pad when UART is in DTE mode.</p> <p>00 <b>ECSPI2_MISO_ALT2</b> — Selecting ALT2 mode of pad ECSPI2_MISO for UART3_RTS_B.</p> <p>01 <b>ECSPI2_SS0_ALT2</b> — Selecting ALT2 mode of pad ECSPI2_SS0 for UART3_CTS_B.</p> <p>10 <b>EPDC_BDR0_ALT2</b> — Selecting ALT2 mode of pad EPDC_BDR0 for UART3_RTS_B.</p> <p>11 <b>EPDC_BDR1_ALT2</b> — Selecting ALT2 mode of pad EPDC_BDR1 for UART3_CTS_B.</p>

### 30.4.505 Select Input Register (IOMUXC\_UART3\_UART\_RX\_DATA\_SELECT\_INPUT)

Address: 20E\_0000h base + 80Ch offset = 20E\_080Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																															DAISY
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_UART3\_UART\_RX\_DATA\_SELECT\_INPUT field descriptions

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**IOMUXC\_UART3\_UART\_RX\_DATA\_SELECT\_INPUT field descriptions (continued)**

Field	Description
2–0 DAISY	<p>Input Select (DAISY) Field</p> <p>Selecting Pads Involved in Daisy Chain.</p> <p>Route RX_DATA signal to pad when UART is in DCE mode. Route TX_DATA signal to pad when UART is in DTE mode.</p> <p>000 <b>AUD_RXC_ALT2</b> — Selecting ALT2 mode of pad AUD_RXC for UART3_TX_DATA.</p> <p>001 <b>AUD_RXFS_ALT2</b> — Selecting ALT2 mode of pad AUD_RXFS for UART3_RX_DATA.</p> <p>010 <b>ECSPI2_MOSI_ALT2</b> — Selecting ALT2 mode of pad ECSPI2_MOSI for UART3_TX_DATA.</p> <p>011 <b>ECSPI2_SCLK_ALT2</b> — Selecting ALT2 mode of pad ECSPI2_SCLK for UART3_RX_DATA.</p> <p>100 <b>EPDC_VCOM0_ALT2</b> — Selecting ALT2 mode of pad EPDC_VCOM0 for UART3_RX_DATA.</p> <p>101 <b>EPDC_VCOM1_ALT2</b> — Selecting ALT2 mode of pad EPDC_VCOM1 for UART3_TX_DATA.</p>

### 30.4.506 Select Input Register (IOMUXC\_UART4\_UART\_RTS\_B\_SELECT\_INPUT)

Address: 20E\_0000h base + 810h offset = 20E\_0810h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																DAISY															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**IOMUXC\_UART4\_UART\_RTS\_B\_SELECT\_INPUT field descriptions**

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value 0.
2–0 DAISY	<p>Input Select (DAISY) Field</p> <p>Selecting Pads Involved in Daisy Chain.</p> <p>Route RTS_B signal to pad when UART is in DCE mode. Route CTS_B signal to pad when UART is in DTE mode.</p> <p>000 <b>AUD_TXD_ALT2</b> — Selecting ALT2 mode of pad AUD_TXD for UART4_CTS_B.</p> <p>001 <b>AUD_TXFS_ALT2</b> — Selecting ALT2 mode of pad AUD_TXFS for UART4_RTS_B.</p> <p>010 <b>KEY_COL7_ALT1</b> — Selecting ALT1 mode of pad KEY_COL7 for UART4_RTS_B.</p> <p>011 <b>KEY_ROW7_ALT1</b> — Selecting ALT1 mode of pad KEY_ROW7 for UART4_CTS_B.</p> <p>100 <b>SD1_DATA6_ALT4</b> — Selecting ALT4 mode of pad SD1_DATA6 for UART4_RTS_B.</p> <p>101 <b>SD1_DATA7_ALT4</b> — Selecting ALT4 mode of pad SD1_DATA7 for UART4_CTS_B.</p>

### 30.4.507 Select Input Register (IOMUXC\_UART4\_UART\_RX\_DATA\_SELECT\_INPUT)

Address: 20E\_0000h base + 814h offset = 20E\_0814h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																DAISY															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_UART4\_UART\_RX\_DATA\_SELECT\_INPUT field descriptions

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value 0.
2–0 DAISY	<p>Input Select (DAISY) Field</p> <p>Selecting Pads Involved in Daisy Chain.</p> <p>Route RX_DATA signal to pad when UART is in DCE mode. Route TX_DATA signal to pad when UART is in DTE mode.</p> <p>000 <b>AUD_RXD_ALT2</b> — Selecting ALT2 mode of pad AUD_RXD for UART4_RX_DATA.            001 <b>AUD_TXC_ALT2</b> — Selecting ALT2 mode of pad AUD_TXC for UART4_TX_DATA.            010 <b>KEY_COL6_ALT1</b> — Selecting ALT1 mode of pad KEY_COL6 for UART4_RX_DATA.            011 <b>KEY_ROW6_ALT1</b> — Selecting ALT1 mode of pad KEY_ROW6 for UART4_TX_DATA.            100 <b>SD1_DATA4_ALT4</b> — Selecting ALT4 mode of pad SD1_DATA4 for UART4_RX_DATA.            101 <b>SD1_DATA5_ALT4</b> — Selecting ALT4 mode of pad SD1_DATA5 for UART4_TX_DATA.            110 <b>UART1_RXD_ALT2</b> — Selecting ALT2 mode of pad UART1_RXD for UART4_RX_DATA.            111 <b>UART1_TXD_ALT2</b> — Selecting ALT2 mode of pad UART1_TXD for UART4_TX_DATA.</p>

### 30.4.508 Select Input Register (IOMUXC\_UART5\_UART\_RTS\_B\_SELECT\_INPUT)

Address: 20E\_0000h base + 818h offset = 20E\_0818h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																DAISY															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_UART5\_UART\_RTS\_B\_SELECT\_INPUT field descriptions

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value 0.
2–0 DAISY	Input Select (DAISY) Field

Table continues on the next page...

**IOMUXC\_UART5\_UART\_RTS\_B\_SELECT\_INPUT field descriptions (continued)**

Field	Description
	Selecting Pads Involved in Daisy Chain.  Route RTS_B signal to pad when UART is in DCE mode. Route CTS_B signal to pad when UART is in DTE mode.
000	<b>ECSPI1_MISO_ALT2</b> — Selecting ALT2 mode of pad ECSPI1_MISO for UART5_RTS_B.
001	<b>ECSPI1_SS0_ALT2</b> — Selecting ALT2 mode of pad ECSPI1_SS0 for UART5_CTS_B.
010	<b>LCD_DATA12_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT12 for UART5_RTS_B.
011	<b>LCD_DATA13_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT13 for UART5_CTS_B.
100	<b>SD2_DATA0_ALT4</b> — Selecting ALT4 mode of pad SD2_DAT0 for UART5_RTS_B.
101	<b>SD2_DATA1_ALT4</b> — Selecting ALT4 mode of pad SD2_DAT1 for UART5_CTS_B.

### 30.4.509 Select Input Register (IOMUXC\_UART5\_UART\_RX\_DATA\_SELECT\_INPUT)

Address: 20E\_0000h base + 81Ch offset = 20E\_081Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_UART5\_UART\_RX\_DATA\_SELECT\_INPUT field descriptions**

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value 0.
2–0 DAISY	Input Select (DAISY) Field  Selecting Pads Involved in Daisy Chain.  Route RX_DATA signal to pad when UART is in DCE mode. Route TX_DATA signal to pad when UART is in DTE mode.  000 <b>ECSPI1_MOSI_ALT2</b> — Selecting ALT2 mode of pad ECSPI1_MOSI for UART5_TX_DATA. 001 <b>ECSPI1_SCLK_ALT2</b> — Selecting ALT2 mode of pad ECSPI1_SCLK for UART5_RX_DATA. 010 <b>LCD_DATA14_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT14 for UART5_RX_DATA. 011 <b>LCD_DATA15_ALT4</b> — Selecting ALT4 mode of pad LCD_DAT15 for UART5_TX_DATA. 100 <b>SD2_DATA2_ALT4</b> — Selecting ALT4 mode of pad SD2_DAT2 for UART5_RX_DATA. 101 <b>SD2_DATA3_ALT4</b> — Selecting ALT4 mode of pad SD2_DAT3 for UART5_TX_DATA. 110 <b>UART1_RXD_ALT4</b> — Selecting ALT4 mode of pad UART1_RXD for UART5_RX_DATA. 111 <b>UART1_TXD_ALT4</b> — Selecting ALT4 mode of pad UART1_TXD for UART5_TX_DATA.

### 30.4.510 Select Input Register (IOMUXC\_USB\_OTG2\_OC\_SELECT\_INPUT)

Address: 20E\_0000h base + 820h offset = 20E\_0820h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_USB\_OTG2\_OC\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>ECSPI1_SCLK_ALT6</b> — Selecting ALT6 mode of pad ECSPI1_SCLK for USB_OTG2_OC. 01 <b>ECSPI2_SCLK_ALT6</b> — Selecting ALT6 mode of pad ECSPI2_SCLK for USB_OTG2_OC. 10 <b>KEY_ROW5_ALT6</b> — Selecting ALT6 mode of pad KEY_ROW5 for USB_OTG2_OC. 11 <b>SD3_DATA2_ALT6</b> — Selecting ALT6 mode of pad SD3_DAT2 for USB_OTG2_OC.

### 30.4.511 Select Input Register (IOMUXC\_USB\_OTG1\_OC\_SELECT\_INPUT)

Address: 20E\_0000h base + 824h offset = 20E\_0824h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_USB\_OTG1\_OC\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>ECSPI2_MISO_ALT6</b> — Selecting ALT6 mode of pad ECSPI2_MISO for USB_OTG1_OC. 01 <b>KEY_ROW4_ALT6</b> — Selecting ALT6 mode of pad KEY_ROW4 for USB_OTG1_OC. 10 <b>SD3_DATA3_ALT6</b> — Selecting ALT6 mode of pad SD3_DAT3 for USB_OTG1_OC.

### 30.4.512 Select Input Register (IOMUXC\_USDHC1\_CARD\_DET\_SELECT\_INPUT)

Address: 20E\_0000h base + 828h offset = 20E\_0828h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_USDHC1\_CARD\_DET\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>ECSPI2_SS0_ALT4</b> — Selecting ALT4 mode of pad ECSPI2_SS0 for SD1_CD_B. 01 <b>FEC_TX_DATA1_ALT3</b> — Selecting ALT3 mode of pad FEC_TXD1 for SD1_CD_B. 10 <b>KEY_COL0_ALT4</b> — Selecting ALT4 mode of pad KEY_COL0 for SD1_CD_B. 11 <b>KEY_ROW7_ALT6</b> — Selecting ALT6 mode of pad KEY_ROW7 for SD1_CD_B.

### 30.4.513 Select Input Register (IOMUXC\_USDHC1\_WP\_ON\_SELECT\_INPUT)

Address: 20E\_0000h base + 82Ch offset = 20E\_082Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_USDHC1\_WP\_ON\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>ECSPI2_MISO_ALT4</b> — Selecting ALT4 mode of pad ECSPI2_MISO for SD1_WP. 01 <b>FEC_TX_EN_ALT3</b> — Selecting ALT3 mode of pad FEC_TX_EN for SD1_WP. 10 <b>KEY_COL7_ALT6</b> — Selecting ALT6 mode of pad KEY_COL7 for SD1_WP. 11 <b>KEY_ROW0_ALT4</b> — Selecting ALT4 mode of pad KEY_ROW0 for SD1_WP.

### 30.4.514 Select Input Register (IOMUXC\_USDHC2\_CARD\_DET\_SELECT\_INPUT)

Address: 20E\_0000h base + 830h offset = 20E\_0830h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														DAISY	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



**IOMUXC\_USDHC2\_CARD\_DET\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>ECSPI1_SS0_ALT4</b> — Selecting ALT4 mode of pad ECSPI1_SS0 for SD2_CD_B. 01 <b>EPDC_GDSP_ALT6</b> — Selecting ALT6 mode of pad EPDC_GDSP for SD2_CD_B. 10 <b>SD2_DATA7_ALT4</b> — Selecting ALT4 mode of pad SD2_DAT7 for SD2_CD_B.

### 30.4.515 Select Input Register (IOMUXC\_USDHC2\_WP\_ON\_SELECT\_INPUT)

Address: 20E\_0000h base + 834h offset = 20E\_0834h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_USDHC2\_WP\_ON\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>ECSPI1_MISO_ALT4</b> — Selecting ALT4 mode of pad ECSPI1_MISO for SD2_WP. 01 <b>EPDC_GDRL_ALT6</b> — Selecting ALT6 mode of pad EPDC_GDRL for SD2_WP. 10 <b>SD2_DATA6_ALT4</b> — Selecting ALT4 mode of pad SD2_DAT6 for SD2_WP.

### 30.4.516 Select Input Register (IOMUXC\_USDHC3\_CARD\_DET\_SELECT\_INPUT)

Address: 20E\_0000h base + 838h offset = 20E\_0838h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_USDHC3\_CARD\_DET\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_PWR_WAKE_ALT6</b> — Selecting ALT6 mode of pad EPDC_PWRWAKEUP for SD3_CD_B. 01 <b>FEC_TX_DATA1_ALT4</b> — Selecting ALT4 mode of pad FEC_TXD1 for SD3_CD_B. 10 <b>I2C2_SDA_ALT4</b> — Selecting ALT4 mode of pad I2C2_SDA for SD3_CD_B. 11 <b>REF_CLK_32K_ALT6</b> — Selecting ALT6 mode of pad REF_CLK_32K for SD3_CD_B.

### 30.4.517 Select Input Register (IOMUXC\_USDHC3\_DATA4\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 83Ch offset = 20E\_083Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_USDHC3\_DATA4\_IN\_SELECT\_INPUT field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_COL1_ALT4</b> — Selecting ALT4 mode of pad KEY_COL1 for SD3_DATA4. 1 <b>SD2_DATA4_ALT1</b> — Selecting ALT1 mode of pad SD2_DAT4 for SD3_DATA4.

### 30.4.518 Select Input Register (IOMUXC\_USDHC3\_DATA5\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 840h offset = 20E\_0840h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_USDHC3\_DATA5\_IN\_SELECT\_INPUT field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_ROW1_ALT4</b> — Selecting ALT4 mode of pad KEY_ROW1 for SD3_DATA5. 1 <b>SD2_DATA5_ALT1</b> — Selecting ALT1 mode of pad SD2_DAT5 for SD3_DATA5.

30.4.519 Select Input Register  
(IOMUXC\_USDHC3\_DATA6\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 844h offset = 20E\_0844h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_USDHC3\_DATA6\_IN\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_COL2_ALT4</b> — Selecting ALT4 mode of pad KEY_COL2 for SD3_DATA6. 1 <b>SD2_DATA6_ALT1</b> — Selecting ALT1 mode of pad SD2_DAT6 for SD3_DATA6.

### 30.4.520 Select Input Register (IOMUXC\_USDHC3\_DATA7\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 848h offset = 20E\_0848h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_USDHC3\_DATA7\_IN\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>KEY_ROW2_ALT4</b> — Selecting ALT4 mode of pad KEY_ROW2 for SD3_DATA7. 1 <b>SD2_DATA7_ALT1</b> — Selecting ALT1 mode of pad SD2_DAT7 for SD3_DATA7.

### 30.4.521 Select Input Register (IOMUXC\_USDHC3\_WP\_ON\_SELECT\_INPUT)

Address: 20E\_0000h base + 84Ch offset = 20E\_084Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_USDHC3\_WP\_ON\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_PWR_STAT_ALT6</b> — Selecting ALT6 mode of pad EPDC_PWRSTAT for SD3_WP. 01 <b>FEC_TX_EN_ALT4</b> — Selecting ALT4 mode of pad FEC_TX_EN for SD3_WP. 10 <b>I2C2_SCL_ALT4</b> — Selecting ALT4 mode of pad I2C2_SCL for SD3_WP. 11 <b>REF_CLK_24M_ALT6</b> — Selecting ALT6 mode of pad REF_CLK_24M for SD3_WP.

### 30.4.522 Select Input Register (IOMUXC\_USDHC4\_CARD\_CLK\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 850h offset = 20E\_0850h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_USDHC4\_CARD\_CLK\_IN\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_BDR0_ALT1</b> — Selecting ALT1 mode of pad EPDC_BDR0 for SD4_CLK. 01 <b>FEC_MDIO_ALT1</b> — Selecting ALT1 mode of pad FEC_MDIO for SD4_CLK. 10 <b>KEY_COL4_ALT4</b> — Selecting ALT4 mode of pad KEY_COL4 for SD4_CLK.

### 30.4.523 Select Input Register (IOMUXC\_USDHC4\_CARD\_DET\_SELECT\_INPUT)

Address: 20E\_0000h base + 854h offset = 20E\_0854h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_USDHC4\_CARD\_DET\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_DATA11_ALT6</b> — Selecting ALT6 mode of pad EPDC_D11 for SD4_CD_B. 1 <b>EPDC_PWR_CTRL3_ALT6</b> — Selecting ALT6 mode of pad EPDC_PWRCTRL3 for SD4_CD_B.

### 30.4.524 Select Input Register (IOMUXC\_USDHC4\_CMD\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 858h offset = 20E\_0858h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_USDHC4\_CMD\_IN\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_BDR1_ALT1</b> — Selecting ALT1 mode of pad EPDC_BDR1 for SD4_CMD. 01 <b>FEC_TX_CLK_ALT1</b> — Selecting ALT1 mode of pad FEC_TX_CLK for SD4_CMD. 10 <b>KEY_ROW4_ALT4</b> — Selecting ALT4 mode of pad KEY_ROW4 for SD4_CMD.

### 30.4.525 Select Input Register (IOMUXC\_USDHC4\_DATA0\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 85Ch offset = 20E\_085Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_USDHC4\_DATA0\_IN\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_PWR_COM_ALT1</b> — Selecting ALT1 mode of pad EPDC_PWRCOM for SD4_DATA0. 01 <b>FEC_RX_ER_ALT1</b> — Selecting ALT1 mode of pad FEC_RX_ER for SD4_DATA0. 10 <b>KEY_COL5_ALT4</b> — Selecting ALT4 mode of pad KEY_COL5 for SD4_DATA0.



### 30.4.526 Select Input Register (IOMUXC\_USDHC4\_DATA1\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 860h offset = 20E\_0860h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_USDHC4\_DATA1\_IN\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_PWR_IRQ_ALT1</b> — Selecting ALT1 mode of pad EPDC_PWRINT for SD4_DATA1. 01 <b>FEC_CRS_DV_ALT1</b> — Selecting ALT1 mode of pad FEC_CRS_DV for SD4_DATA1. 10 <b>KEY_ROW5_ALT4</b> — Selecting ALT4 mode of pad KEY_ROW5 for SD4_DATA1.

### 30.4.527 Select Input Register (IOMUXC\_USDHC4\_DATA2\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 864h offset = 20E\_0864h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_USDHC4\_DATA2\_IN\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**IOMUXC\_USDHC4\_DATA2\_IN\_SELECT\_INPUT field descriptions (continued)**

Field	Description
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_PWR_STAT_ALT1</b> — Selecting ALT1 mode of pad EPDC_PWRSTAT for SD4_DATA2. 01 <b>FEC_RX_DATA1_ALT1</b> — Selecting ALT1 mode of pad FEC_RXD1 for SD4_DATA2. 10 <b>KEY_COL6_ALT4</b> — Selecting ALT4 mode of pad KEY_COL6 for SD4_DATA2.

### 30.4.528 Select Input Register (IOMUXC\_USDHC4\_DATA3\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 868h offset = 20E\_0868h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_USDHC4\_DATA3\_IN\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>EPDC_PWR_WAKE_ALT1</b> — Selecting ALT1 mode of pad EPDC_PWRWAKEUP for SD4_DATA3. 01 <b>FEC_TX_DATA0_ALT1</b> — Selecting ALT1 mode of pad FEC_TXD0 for SD4_DATA3. 10 <b>KEY_ROW6_ALT4</b> — Selecting ALT4 mode of pad KEY_ROW6 for SD4_DATA3.

### 30.4.529 Select Input Register (IOMUXC\_USDHC4\_DATA4\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 86Ch offset = 20E\_086Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_USDHC4\_DATA4\_IN\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>FEC_MDC_ALT1</b> — Selecting ALT1 mode of pad FEC_MDC for SD4_DATA4. 01 <b>KEY_COL7_ALT4</b> — Selecting ALT4 mode of pad KEY_COL7 for SD4_DATA4. 10 <b>LCD_CLK_ALT1</b> — Selecting ALT1 mode of pad LCD_CLK for SD4_DATA4.

### 30.4.530 Select Input Register (IOMUXC\_USDHC4\_DATA5\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 870h offset = 20E\_0870h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_USDHC4\_DATA5\_IN\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**IOMUXC\_USDHC4\_DATA5\_IN\_SELECT\_INPUT field descriptions (continued)**

Field	Description
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>FEC_RX_DATA0_ALT1</b> — Selecting ALT1 mode of pad FEC_RXD0 for SD4_DATA5. 01 <b>KEY_ROW7_ALT4</b> — Selecting ALT4 mode of pad KEY_ROW7 for SD4_DATA5. 10 <b>LCD_ENABLE_ALT1</b> — Selecting ALT1 mode of pad LCD_ENABLE for SD4_DATA5.

### 30.4.531 Select Input Register (IOMUXC\_USDHC4\_DATA6\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 874h offset = 20E\_0874h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_USDHC4\_DATA6\_IN\_SELECT\_INPUT field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>FEC_TX_EN_ALT1</b> — Selecting ALT1 mode of pad FEC_TX_EN for SD4_DATA6. 01 <b>KEY_COL3_ALT4</b> — Selecting ALT4 mode of pad KEY_COL3 for SD4_DATA6. 10 <b>LCD_HSYNC_ALT1</b> — Selecting ALT1 mode of pad LCD_HSYNC for SD4_DATA6.

### 30.4.532 Select Input Register (IOMUXC\_USDHC4\_DATA7\_IN\_SELECT\_INPUT)

Address: 20E\_0000h base + 878h offset = 20E\_0878h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### IOMUXC\_USDHC4\_DATA7\_IN\_SELECT\_INPUT field descriptions

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  00 <b>FEC_TX_DATA1_ALT1</b> — Selecting ALT1 mode of pad FEC_TXD1 for SD4_DATA7. 01 <b>KEY_ROW3_ALT4</b> — Selecting ALT4 mode of pad KEY_ROW3 for SD4_DATA7. 10 <b>LCD_VSYNC_ALT1</b> — Selecting ALT1 mode of pad LCD_VSYNC for SD4_DATA7.

### 30.4.533 Select Input Register (IOMUXC\_USDHC4\_WP\_ON\_SELECT\_INPUT)

Address: 20E\_0000h base + 87Ch offset = 20E\_087Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_USDHC4\_WP\_ON\_SELECT\_INPUT field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_DATA10_ALT6</b> — Selecting ALT6 mode of pad EPDC_D10 for SD4_WP. 1 <b>EPDC_PWR_CTRL2_ALT6</b> — Selecting ALT6 mode of pad EPDC_PWRCTRL2 for SD4_WP.

### 30.4.534 Select Input Register (IOMUXC\_EIM\_DTACK\_B\_SELECT\_INPUT)

Address: 20E\_0000h base + 880h offset = 20E\_0880h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**IOMUXC\_EIM\_DTACK\_B\_SELECT\_INPUT field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_PWR_WAKE_ALT3</b> — Selecting ALT3 mode of pad EPDC_PWRWAKEUP for EIM_DTACK_B. 1 <b>LCD_RESET_ALT1</b> — Selecting ALT1 mode of pad LCD_RESET for EIM_DTACK_B.

### 30.4.535 Select Input Register (IOMUXC\_EIM\_WAIT\_B\_SELECT\_INPUT)

Address: 20E\_0000h base + 884h offset = 20E\_0884h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															DAISY
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

IOMUXC\_EIM\_WAIT\_B\_SELECT\_INPUT field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 DAISY	Input Select (DAISY) Field Selecting Pads Involved in Daisy Chain.  0 <b>EPDC_PWR_STAT_ALT3</b> — Selecting ALT3 mode of pad EPDC_PWRSTAT for EIM_WAIT_B. 1 <b>LCD_RESET_ALT3</b> — Selecting ALT3 mode of pad LCD_RESET for EIM_WAIT_B.







## **Chapter 31**

# **Keypad Port (KPP)**

## 31.1 Overview

The Keypad Port (KPP) is a 16-bit peripheral that can be used as a keypad matrix interface or as general purpose input/output (I/O).

The figure below shows the KPP block diagram. The KPP provides interface for the keypad matrix with 2-point contact or 3-point contact keys. The KPP is designed to simplify the software task of scanning a keypad matrix. With appropriate software support, the KPP is capable of detecting, debouncing, and decoding one or multiple keys pressed simultaneously on the keypad.

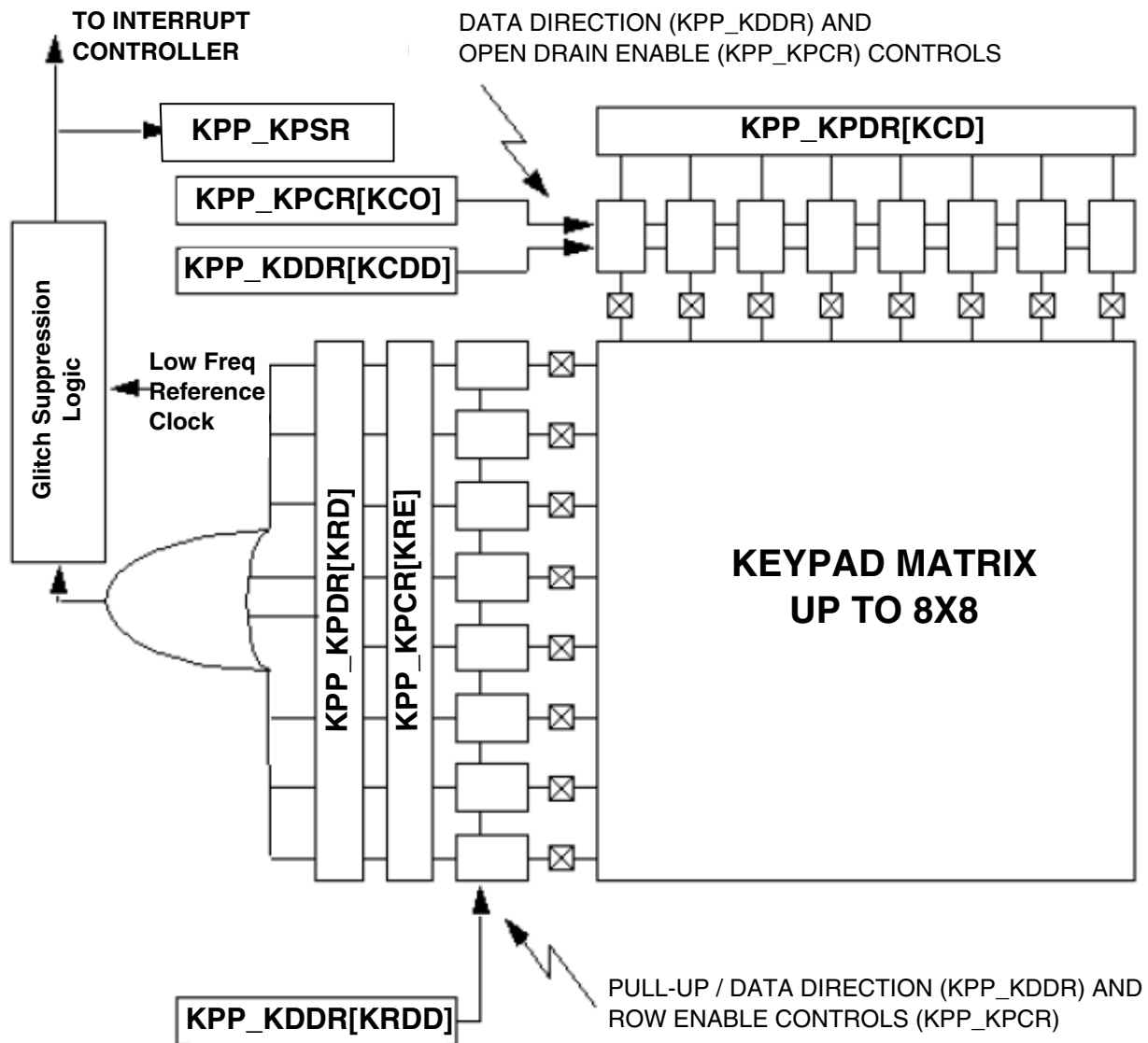


Figure 31-1. KPP Peripheral Block Diagram

### 31.1.1 Features

The KPP includes these distinctive features:

- Supports up to an 8 x 8 external key pad matrix
- Port pins can be used as general purpose I/O
- Open drain design
- Glitch suppression circuit design
- Multiple-key detection
- Long key-press detection
- Standby key-press detection
- Synchronizer chain clear
- Supports a 2-point and 3-point contact key matrix

### 31.1.2 Modes and Operations

This block supports the following modes:

- Run Mode-This is the normal functional mode in which the KPP can detect any key press event.
- Low Power Mode-The keypad can detect any key press even in low power modes (when there is no MCU clock).

## 31.2 Clocks

The table found here describes the clock sources for KPP.

Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 31-1. KPP Clocks**

Clock name	Clock Root	Description
ipg_clk_32k	ckil_sync_clk_root	Low-frequency reference clock (32kHz)
ipg_clk_s	ipg_clk_root	Peripheral access clock

## 31.3 External Signals

There are several pins dedicated to the KPP. Keypads of any configuration up to eight rows and eight columns are supported through the software configuration of the peripheral pins. Any pins not used for the keypad are available as general purpose I/O. The registers are configured such that the pins can be treated as an I/O port up to 16 bits wide.

See the table below for the list of external signals.

**Table 31-2. KPP External Signals**

Signal	Description	Pad	Mode	Direction
KEY_COL0	Column input or output pin, from chip	KEY_COL0	ALT0	I/O
		LCD_DAT8	ALT1	
		SD1_CLK	ALT2	
KEY_COL1	Column input or output pin, from chip	KEY_COL1	ALT0	I/O
		LCD_DAT10	ALT1	
		SD1_DAT0	ALT2	
KEY_COL2	Column input or output pin, from chip	KEY_COL2	ALT0	I/O
		LCD_DAT12	ALT1	
		SD1_DAT2	ALT2	
KEY_COL3	Column input or output pin, from chip	KEY_COL3	ALT0	I/O
		LCD_DAT14	ALT1	
		SD1_DAT4	ALT2	
KEY_COL4	Column input or output pin, from chip	KEY_COL4	ALT0	I/O
		LCD_DAT16	ALT1	
		SD1_DAT6	ALT2	
KEY_COL5	Column input or output pin, from chip	KEY_COL5	ALT0	I/O
		LCD_DAT18	ALT1	
		SD3_CLK	ALT2	
KEY_COL6	Column input or output pin, from chip	KEY_COL6	ALT0	I/O
		LCD_DAT20	ALT1	
		SD3_DAT0	ALT2	
KEY_COL7	Column input or output pin, from chip	KEY_COL7	ALT0	I/O
		LCD_DAT22	ALT1	
		SD3_DAT2	ALT2	
KEY_ROW0	Row input or output pin, from chip	KEY_ROW0	ALT0	I/O
		LCD_DAT9	ALT1	
		SD1_CMD	ALT2	
KEY_ROW1	Row input or output pin, from chip	KEY_ROW1	ALT0	I/O
		LCD_DAT11	ALT1	
		SD1_DAT1	ALT2	

*Table continues on the next page...*

**Table 31-2. KPP External Signals (continued)**

Signal	Description	Pad	Mode	Direction
KEY_ROW2	Row input or output pin, from chip	KEY_ROW2	ALT0	I/O
		LCD_DAT13	ALT1	
		SD1_DAT3	ALT2	
KEY_ROW3	Row input or output pin, from chip	KEY_ROW3	ALT0	I/O
		LCD_DAT15	ALT1	
		SD1_DAT5	ALT2	
KEY_ROW4	Row input or output pin, from chip	KEY_ROW4	ALT0	I/O
		LCD_DAT17	ALT1	
		SD1_DAT7	ALT2	
KEY_ROW5	Row input or output pin, from chip	KEY_ROW5	ALT0	I/O
		LCD_DAT19	ALT1	
		SD3_CMD	ALT2	
KEY_ROW6	Row input or output pin, from chip	KEY_ROW6	ALT0	I/O
		LCD_DAT21	ALT1	
		SD3_DAT1	ALT2	
KEY_ROW7	Row input or output pin, from chip	KEY_ROW7	ALT0	I/O
		LCD_DAT23	ALT1	
		SD3_DAT3	ALT2	

### 31.3.1 Input Pins

Any of the 16 pins associated with the KPP can be configured as inputs by writing a "0" to the appropriate bits in the KPP\_KDDR. Additionally, the least significant 8 bits (ROW inputs) corresponding to KPP\_KDDR[KRDD] have internal pull-ups, which are enabled when the pin is used as an input.

### 31.3.2 Output Pins

Any of the 16 pins associated with the KPP can be configured as outputs by writing the appropriate bits in the KPP\_KDDR to a "1". Additionally, the 8 most significant bits (15-8) can be designated as open drain outputs by writing a "1" to the appropriate bits in the KPP\_KPCR. The lower 8 bits (7-0) are always in "totem pole" style, driven when configured as outputs.

See the table below.

**Table 31-3. Keypad Port Column Modes**

KPP_KDDR (15:8)	KPP_KPCR (15:8)	Pin Function
0	x	Input
1	0	Totem-Pole Output
1	1	Open-Drain Output

### NOTE

Totem pole capability should be provided for column pins. Totem pole configuration helps for a faster discharge of keypad capacitance when all columns need to be quickly brought to a "1" during the scan routine. With this configuration, delay between the scanning of two subsequent columns is reduced.

### 31.3.3 Generation of Transfer Error Signal on Peripheral Bus

If there is an access to an address which is not implemented, then the KPP asserts a transfer error signal on Peripheral Bus.

## 31.4 Functional Description

The Keypad Port (KPP) is designed to simplify the software task of scanning a keypad matrix. With appropriate software support and matrix organization, the KPP is capable of detecting, debouncing, and decoding one or more keys pressed simultaneously on the keypad.

Logic in the KPP is capable of detecting a key press even while the processor is in one of the low power standby modes provided that a low frequency reference clock is on. The KPP may generate an ARM platform interrupt any time a key press or key release is detected. This interrupt is capable of forcing the processor out of a low power mode.

### 31.4.1 Keypad Matrix Construction

The KPP is designed to interface to a keypad matrix, which shorts the intersecting row and column lines together whenever a key is depressed. The interface is not optimized for any other switch configuration.

### 31.4.2 Keypad Port Configuration

The software must initialize the KPP for the size of the keypad matrix. Pins connected to the keypad columns should be configured as open-drain outputs. Pins connected to the keypad rows should be configured as inputs. On-chip, pull-up resistors should be implemented for active keypad rows.

In addition to enabled row inputs in the Keypad Control register, corresponding interrupt (depress or/and release) must also be enabled to generate an interrupt.

Discrete switches that are not part of the matrix may be connected to any unused row inputs. The second terminal of the discrete switch is connected to ground. The hardware detects closures of these switches without the need for software polling.

### 31.4.3 Keypad Matrix Scanning

Keypad scanning is performed by a software loop that walks a zero across each of the keypad columns, reading the value on the rows at each step. The process is repeated several times in succession, with the results of each pass optionally compared to those from the previous pass. When several (3 or 4) consecutive scans yield the same key closures, a valid key press has been detected. Software then can decode exactly which switch was depressed and pass the value up to the next higher software layer.

The basic debouncing period, which must be defined in the software routine, may be controlled with an internal timer. The basic period is the period between the scan of two consecutive columns, so the debouncing time between two consecutive scans of the whole matrix shall be the number of columns multiplied by the basic period.

### 31.4.4 Keypad Standby

There is no need for the ARM platform to continually scan the keypad. Between key presses, the keypad can be left in a state that requires no software intervention until the next key press is detected. To place the keypad in a standby state, software should write all column outputs low. Row inputs are left enabled. At this point, the ARM platform can attend to other tasks or revert to a low power standby mode. The KPP will interrupt the ARM platform if any key is pressed.

## Functional Description

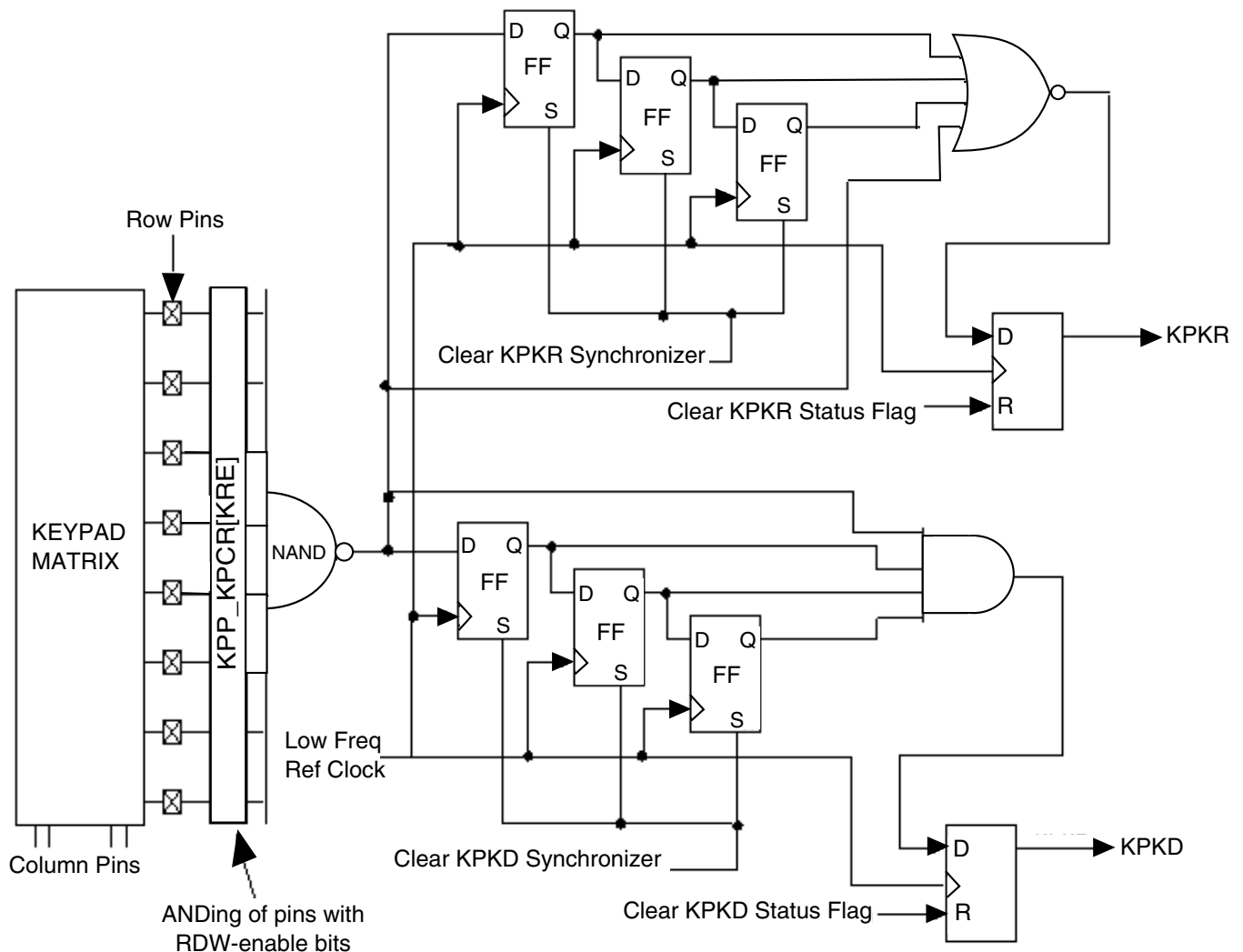
Upon receiving a keypad interrupt, the ARM platform should set all the column strobes high, and begin a normal keypad scanning routine to determine which key was pressed. It is important that open-drain drivers be used when scanning to prevent a possible DC path between power and ground through two or more switches.



### 31.4.5 Glitch Suppression on Keypad Inputs

A glitch suppression circuit qualifies the keypad inputs to prevent noise from inadvertently interrupting the ARM platform. The circuit is a 4-state synchronizer clocked from a low frequency reference clock source.

This clock must continue to run in any low power mode where the keypad is a wake-up source, as the ARM platform interrupt is generated from the synchronized input. An interrupt is not generated until all four synchronizer stages have latched a valid key assertion. This guarantees the filtering out of any noise less than three clock periods in duration of a low frequency reference clock. Noise filtering of the duration between three to four clock periods cannot be guaranteed. The interrupt output is latched in an S-R latch and remains asserted until cleared by the software. The Set input of the latch is rising-edge clocked. See the figure below.

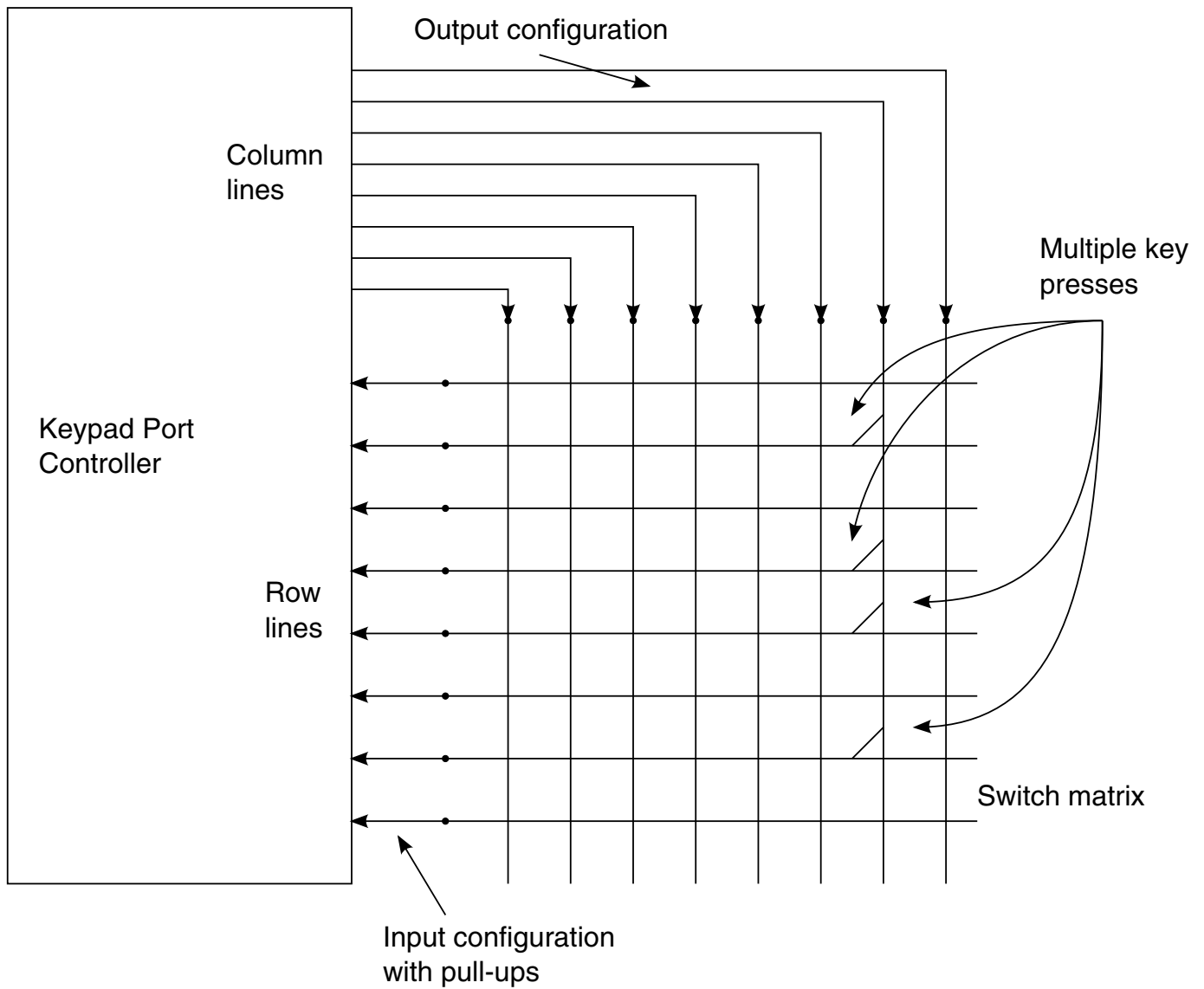


**Figure 31-2. Keypad Synchronizer Functional Diagram**

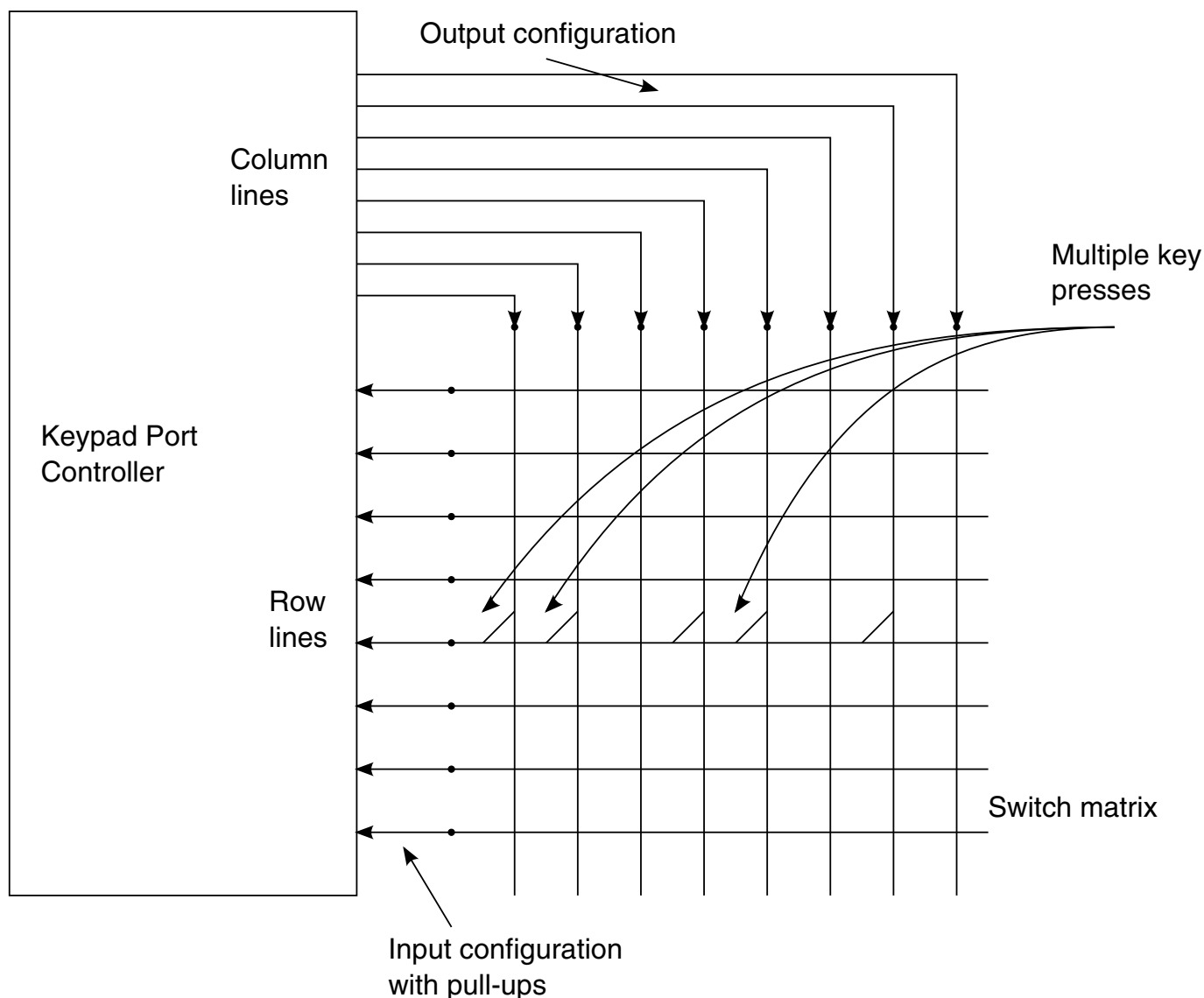
### 31.4.6 Multiple Key Closures

Using the key press and Key release interrupts, the software can detect multiple keys or achieve n key rollover. The key scanning routine can be programmed accordingly.

See the following figures for illustrations of the interfacing of a 2-contact keypad matrix with the KPP controller. With proper enabling of row lines and the performing scan-routine, multiple key presses can be detected. When keys present on the same row are pressed, corresponding row lines (multiple lines) become low when the column is driven low during a scan-routine. By reading the data-register, pressed keys can be detected. Similarly, when keys present on same row line are pressed, the corresponding row line (only one line) becomes low when logic "0" is driven on the column line during a scan-routine.



**Figure 31-3. Multiple Key Presses on Same Column Line (Simplified View)**



**Figure 31-4. Multiple Key Presses on Same Row Line (Simplified View)**

#### NOTE

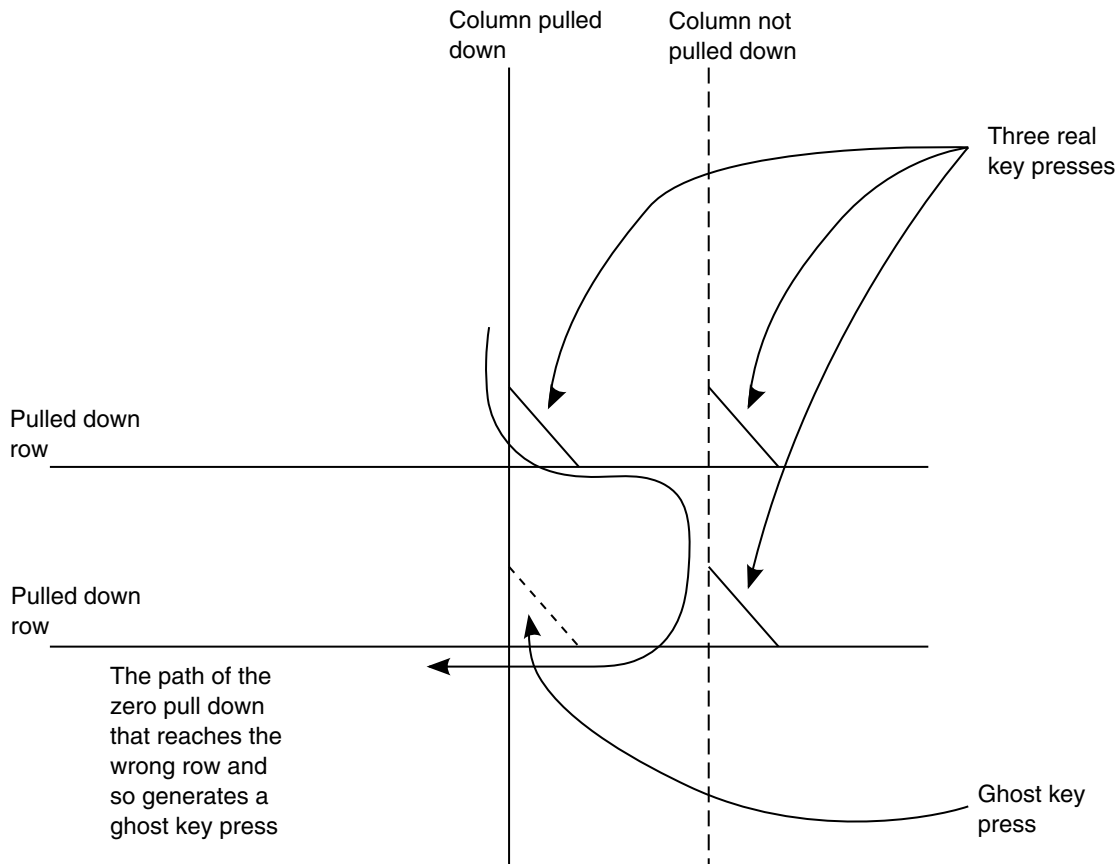
An n key rollover is a technique with which the system can recognize the order in which keys are pressed.

### 31.4.6.1 Ghost Key Problem and Correction

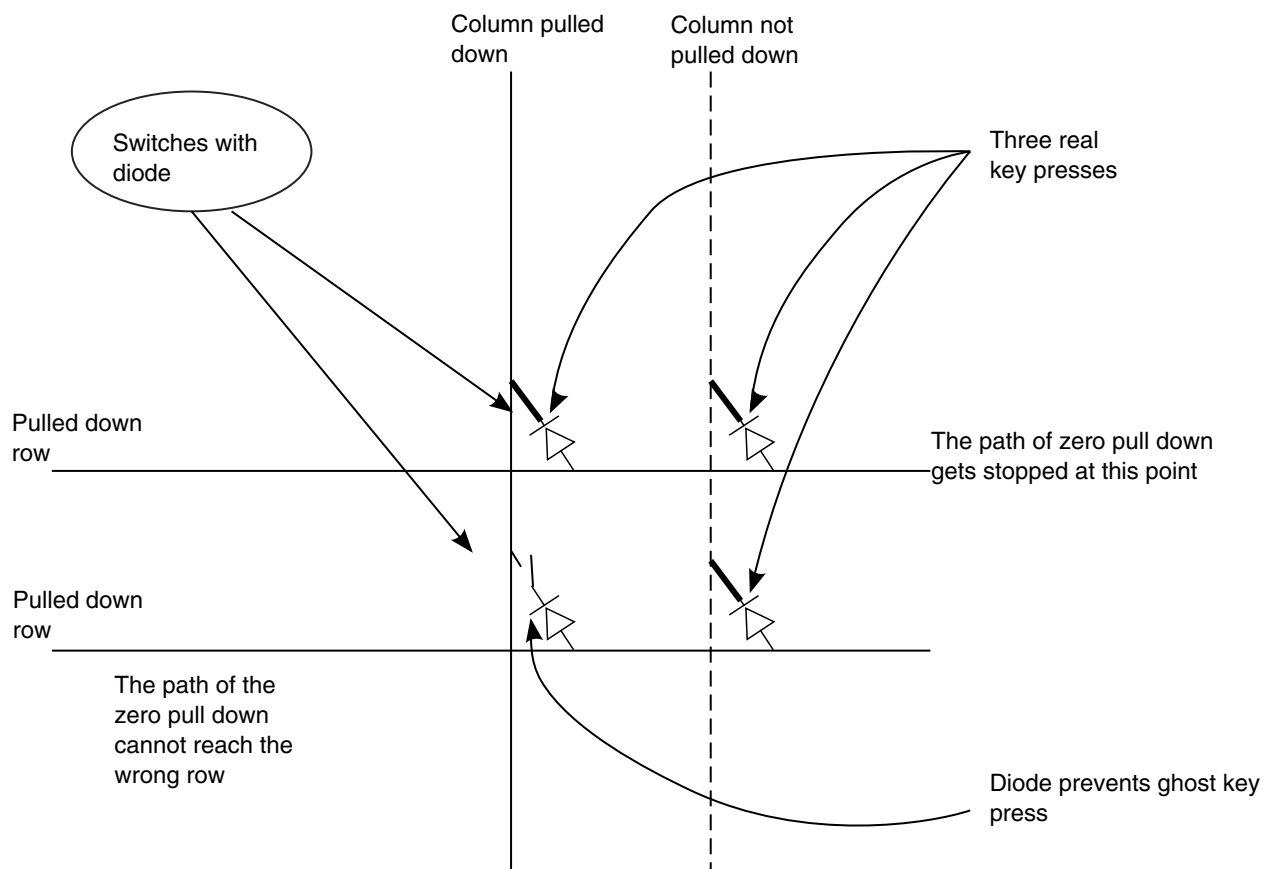
The KPP detects if one or multiple keys are pressed or released. In the case where a simple keypad matrix with two-contact switches is used, there is a chance of "ghost" key detection when three or more keys are pressed. This is a limitation imposed by such a keypad matrix.

As can be seen in [Figure 31-5](#), three keys pressed simultaneously can cause a short between the column currently "scanned" by the software and another column. Depending on the location of the third key pressed, a "ghost" key press may be detected.

However, this can be corrected by using a keypad matrix that provides "ghost" key protection. Such a matrix implements a one-way "diode" at all keypad points between rows and columns. This way, the multiple pressing of three keys will not cause a short at a fourth key (see [Figure 31-6](#)).



**Figure 31-5. Decoding Wrong Three- Key-Presses**



**Figure 31-6. Matrix with "Ghost" Key Protections**

### 31.4.7 3-Point Contact Keys Support

The KPP supports interfacing to a matrix consisting of 3-point contact keys. As shown in [Figure 31-7](#), two points of such a key are connected to keypad lines, while a third point is connected to ground (low logic).

The keypad lines should be configured as input and a pull-up should be present on these lines. When such a key is pressed, corresponding keypad lines go low and an interrupt is generated. There is no need to perform a scanning routine for identification of pressed key as it can be done by reading the keypad data-register. A limitation with such a matrix is that for every key at least one keypad row line should be used.

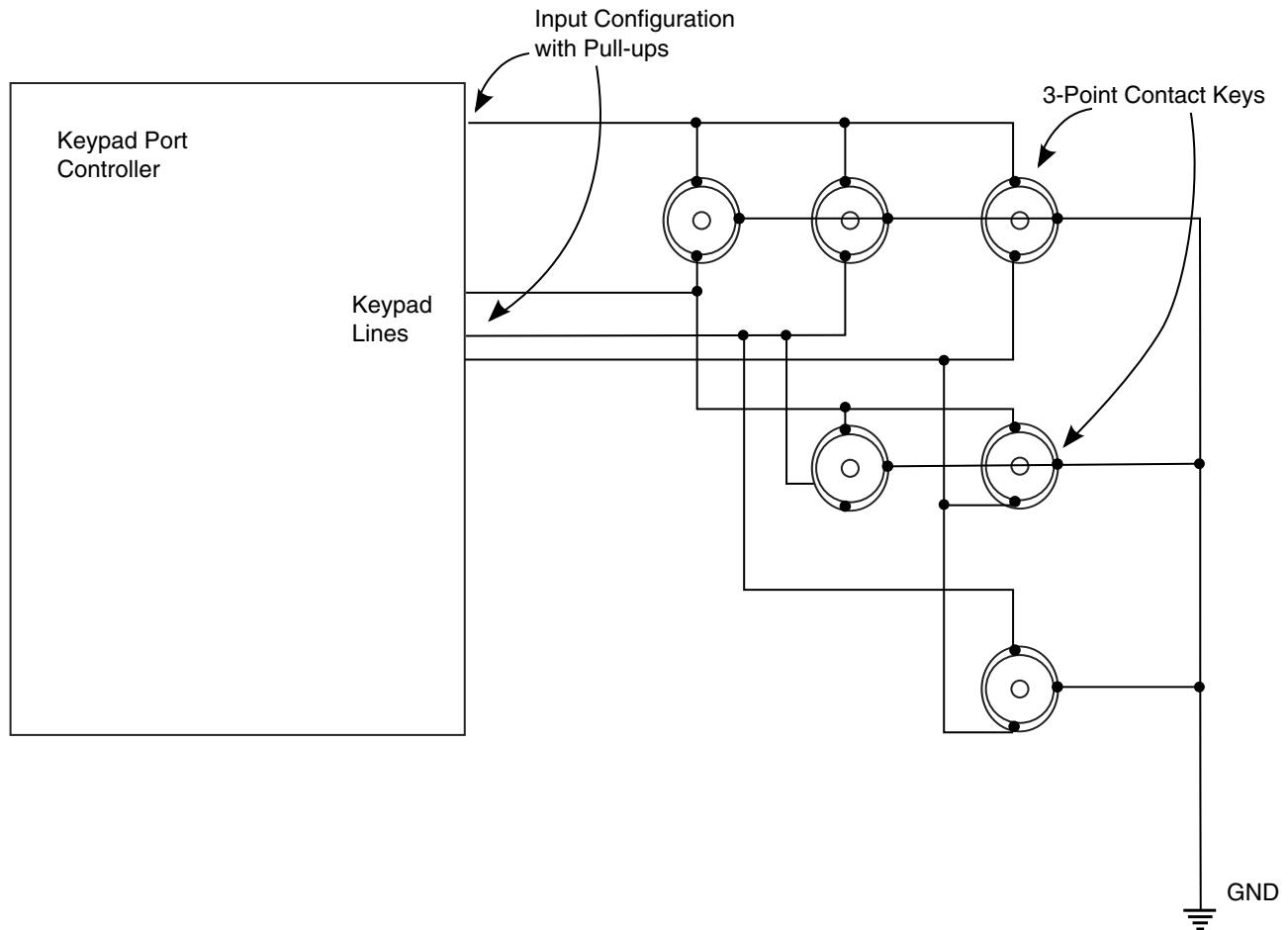


Figure 31-7. KPP Interface with 3-point Contact Key Matrix (Simplified View)

## 31.5 Initialization/Application Information

### 31.5.1 Typical Keypad Configuration and Scanning Sequence

Perform the following steps to configure the keypad:

1. Enable the number of rows in the keypad (KPP\_KPCR[KRE]).
2. Write 0s to KPP\_KPDR[KCD].
3. Configure the keypad columns as open-drain (KPP\_KPCR[KCO]).
4. Configure columns as output (KPP\_KDDR[KCDD]) and rows as input (KPP\_KDDR[KRDD]).
5. Clear the KPKD Status Flag and Synchronizer chain.
6. Set the KDIE control bit, and clear the KRIE control bit (avoid false release events).
7. (The system is now in standby mode, and awaiting a key press.)

### 31.5.2 Key Press Interrupt Scanning Sequence

Perform the following steps to perform a keypad scanning routine:

1. Disable both (depress and release) keypad interrupts.
2. Write 1s to KPP\_KPDR[KCD], setting column data to 1s.
3. Configure columns as totem pole outputs (for quick discharging of keypad capacitance).
4. Configure columns as open-drain.
5. Write a single column to 0, and other columns to 1.
6. Sample row inputs and save data. Multiple key presses can be detected on a single column.
7. Repeat Steps 2-6 for remaining columns.
8. Return all columns to 0 in preparation for standby mode.
9. Clear KPKD and KPKR status bit(s) by writing to a "1"; set the KPKR synchronizer chain by writing a "1" to the KPP\_KRSS register; and clear the KPKD synchronizer chain by writing a "1" to the KDSC register.
10. Re-enable the appropriate keypad interrupt(s) so that the KDIE detects a key hold condition, or the KRIE detects a key-release event.

### 31.5.3 Additional Comments

The order of key press detection can be done in software only. Therefore, the software may need to run the scan routines at very short intervals of time per the application's demands. The reason that such functionality cannot be put in the KPP is that the block is limited by the number of external pins.



For the keys that require a very precise order (such as game keys), individual GPIO pins may be more useful.

## 31.6 KPP Memory Map/Register Definition

The KPP contains four registers.

**KPP memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20B_8000	Keypad Control Register (KPP_KPCR)	16	R/W	0000h	<a href="#">31.6.1/1989</a>
20B_8002	Keypad Status Register (KPP_KPSR)	16	R/W	0400h	<a href="#">31.6.2/1990</a>
20B_8004	Keypad Data Direction Register (KPP_KDDR)	16	R/W	0000h	<a href="#">31.6.3/1992</a>
20B_8006	Keypad Data Register (KPP_KPDR)	16	R/W	0000h	<a href="#">31.6.4/1992</a>

### 31.6.1 Keypad Control Register (KPP\_KPCR)

The Keypad Control Register determines which of the eight possible column strobes are to be open drain when configured as outputs, and which of the eight row sense lines are considered in generating an interrupt to the core.

It is up to the programmer to ensure that pins being used for functions other than the keypad are properly disabled. The KPP\_KPCR register is byte- or half-word-addressable.

Address: 20B\_8000h base + 0h offset = 20B\_8000h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	KCO								KRE							
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**KPP\_KPCR field descriptions**

Field	Description
15–8 KCO	Keypad Column Strobe Open-Drain Enable. Setting a column open-drain enable bit (KCO7-KCO0) disables the pull-up driver on that pin. Clearing the bit allows the pin to drive to the high state. This bit has no effect when the pin is configured as an input.  <b>NOTE:</b> Configuration of external port control logic (for example, IOMUX) should be done properly so that the KPP controls an open-drain enable of the pin.  0 <b>TOTEM_POLE</b> — Column strobe output is totem pole drive. 1 <b>OPEN_DRAIN</b> — Column strobe output is open drain.

*Table continues on the next page...*

**KPP\_KPCR field descriptions (continued)**

Field	Description
7–0 KRE	Keypad Row Enable. Setting a row enable control bit in this register enables the corresponding row line to participate in interrupt generation. Likewise, clearing a bit disables that row from being used to generate an interrupt. This register is cleared by a reset, disabling all rows. The row-enable logic is independent of the programmed direction of the pin. Writing a "0" to the data register of the pins configured as outputs will cause a keypad interrupt to be generated if the row enable associated with that bit is set.  0 Row is not included in the keypad key press detect. 1 Row is included in the keypad key press detect.

**31.6.2 Keypad Status Register (KPP\_KPSR)**

The Keypad Status Register reflects the state of the key press detect circuit. The KPP\_KPSR register is byte- or half-word-addressable.

Address: 20B\_8000h base + 2h offset = 20B\_8002h

Bit	15	14	13	12	11	10	9	8
Read	0						KRIE	KDIE
Write								
Reset	0	0	0	0	0	1	0	0
Bit	7	6	5	4	3	2	1	0
Read	0				0	0	KPKR	KPKD
Write					KRSS	KDSC	w1c	w1c
Reset	0	0	0	0	0	0	0	0

**KPP\_KPSR field descriptions**

Field	Description
15–10 Reserved	This read-only field is reserved and always has the value 0.
9 KRIE	Keypad Release Interrupt Enable. The software should ensure that the interrupt for a Key Release event is masked until it has entered the key pressed state, and vice versa, unless this activity is desired (as might be the case when a repeated interrupt is to be generated). The synchronizer chains are capable of being initialized to detect repeated key presses or releases. If they are not initialized when the corresponding event flag is cleared, false interrupts may be generated for depress (or release) events shorter than the length of the corresponding chain.  0 No interrupt request is generated when KPKR is set. 1 An interrupt request is generated when KPKR is set.
8 KDIE	Keypad Key Depress Interrupt Enable. Software should ensure that the interrupt for a Key Release event is masked until it has entered the key pressed state, and vice-versa, unless this activity is desired (as might be the case when a repeated interrupt is to be generated). The synchronizer chains are capable of being initialized to detect repeated key presses or releases. If they are not initialized when the corresponding event flag is cleared, false interrupts may be generated for depress (or release) events shorter than the length of the corresponding chain.

*Table continues on the next page...*

**KPP\_KPSR field descriptions (continued)**

<b>Field</b>	<b>Description</b>
	<p>0 No interrupt request is generated when KPKD is set.</p> <p>1 An interrupt request is generated when KPKD is set.</p>
7–4 Reserved	This read-only field is reserved and always has the value 0.
3 KRSS	<p>Key Release Synchronizer Set. Self-clear bit. The Key release synchronizer is set by writing a logic one into this bit.</p> <p>Reads return a value of "0".</p> <p>0 No effect</p> <p>1 Set bits which sets keypad release synchronizer chain</p>
2 KDSC	<p>Key Depress Synchronizer Clear. Self-clear bit. The Key depress synchronizer is cleared by writing a logic "1" into this bit.</p> <p>Reads return a value of "0".</p> <p>0 No effect</p> <p>1 Set bits that clear the keypad depress synchronizer chain</p>
1 KPKR	<p>Keypad Key Release. The keypad key release (KPKR) status bit is set when all enabled rows are detected high after synchronization (the KPKR status bit will be set when cleared by a reset). The KPKR bit may be used to generate a maskable key release interrupt. The key release synchronizer may be set high by software after scanning the keypad to ensure a known state. Due to the logic function of the release and depress synchronizer chains, it is possible to see the re-assertion of a status flag (KPKD or KPKR) if it is cleared by software prior to the system exiting the state it represents.</p> <p>Reset value of register is "0" as long as reset is asserted. However when reset is de-asserted, the value of the register depends upon the external row pins and can become "1".</p> <p>0 No key release detected</p> <p>1 All keys have been released</p>
0 KPKD	<p>Keypad Key Depress. The keypad key depress (KPKD) status bit is set when one or more enabled rows are detected low after synchronization. The KPKD status bit remains set until cleared by the software. The KPKD bit may be used to generate a maskable key depress interrupt. If desired, the software may clear the key press synchronizer chain to allow a repeated interrupt to be generated while a key remains pressed. In this case, a new interrupt will be generated after the synchronizer delay (4 cycles of the low frequency reference clock elapses if a key remains pressed. This functionality can be used to detect a long key press. This allows detection of additional key presses of the same key or other keys.</p> <p>Due to the logic function of the release and depress synchronizer chains, it is possible to see the re-assertion of a status flag (KPKD or KPKR) if it is cleared by the software prior to the system exiting the state it represents.</p> <p>0 No key presses detected</p> <p>1 A key has been depressed</p>

### 31.6.3 Keypad Data Direction Register (KPP\_KDDR)

The bits in the KPP\_KDDR control the direction of the keypad port pins. The upper eight bits in the register affect the pins designated as column strobes, while the lower eight bits affect the row sense pins. Setting any bit in this register configures the corresponding pin as an output. Clearing any bit in this register configures the corresponding port pin as an input. For the Keypad Row DDR, an internal pull-up is enabled if the corresponding bit is clear. This register is cleared by a reset, configuring all pins as inputs. The KPP\_KDDR register is byte- or half-word addressable.

#### NOTE

When a pin is used as row pin for keypad purposes, all corresponding pull-ups should be enabled at the upper level (for example, IOMUX) when the bit in KRDD is cleared.

Address: 20B\_8000h base + 4h offset = 20B\_8004h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	KCDD								KRDD							
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### KPP\_KDDR field descriptions

Field	Description
15–8 KCDD	Keypad Column Data Direction Register. Setting a bit configures the corresponding COL $n$ pin as an output (where $n = 7$ through 0).  0 <b>INPUT</b> — COL $n$ pin is configured as an input. 1 <b>OUTPUT</b> — COL $n$ pin is configured as an output.
7–0 KRDD	Keypad Row Data Direction. Setting a bit configures the corresponding ROW $n$ pin as an output (where $n = 7$ through 0).  0 <b>INPUT</b> — ROW $n$ pin configured as an input. 1 <b>OUTPUT</b> — ROW $n$ pin configured as an output.

### 31.6.4 Keypad Data Register (KPP\_KPDR)

This 16-bit register is used to access the column and row data. Data written to this register is stored in an internal latch, and for each pin configured as an output, the stored data is driven onto the pin. A read of this register returns the value on the pin for those bits configured as inputs. Otherwise, the value read is the value stored in the register.

The KPP\_KPDR register is byte- or half-word addressable. This register is not initialized by a reset. Valid data should be written to this register before any bits are configured as outputs.

Address: 20B\_8000h base + 6h offset = 20B\_8006h

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	KCD								KRD							
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### KPP\_KPDR field descriptions

Field	Description
15–8 KCD	Keypad Column Data. A read of these bits returns the value on the pin for those bits configured as inputs. Otherwise, the value read is the value stored in the register. 0 Read/Write "0" from/to column ports 1 Read/Write "1" from/to column ports
7–0 KRD	Keypad Row Data. A read of these bits returns the value on the pin for those bits configured as inputs. Otherwise, the value read is the value stored in the register. 0 Read/Write "0" from/to row ports 1 Read/Write "1" from/to row ports



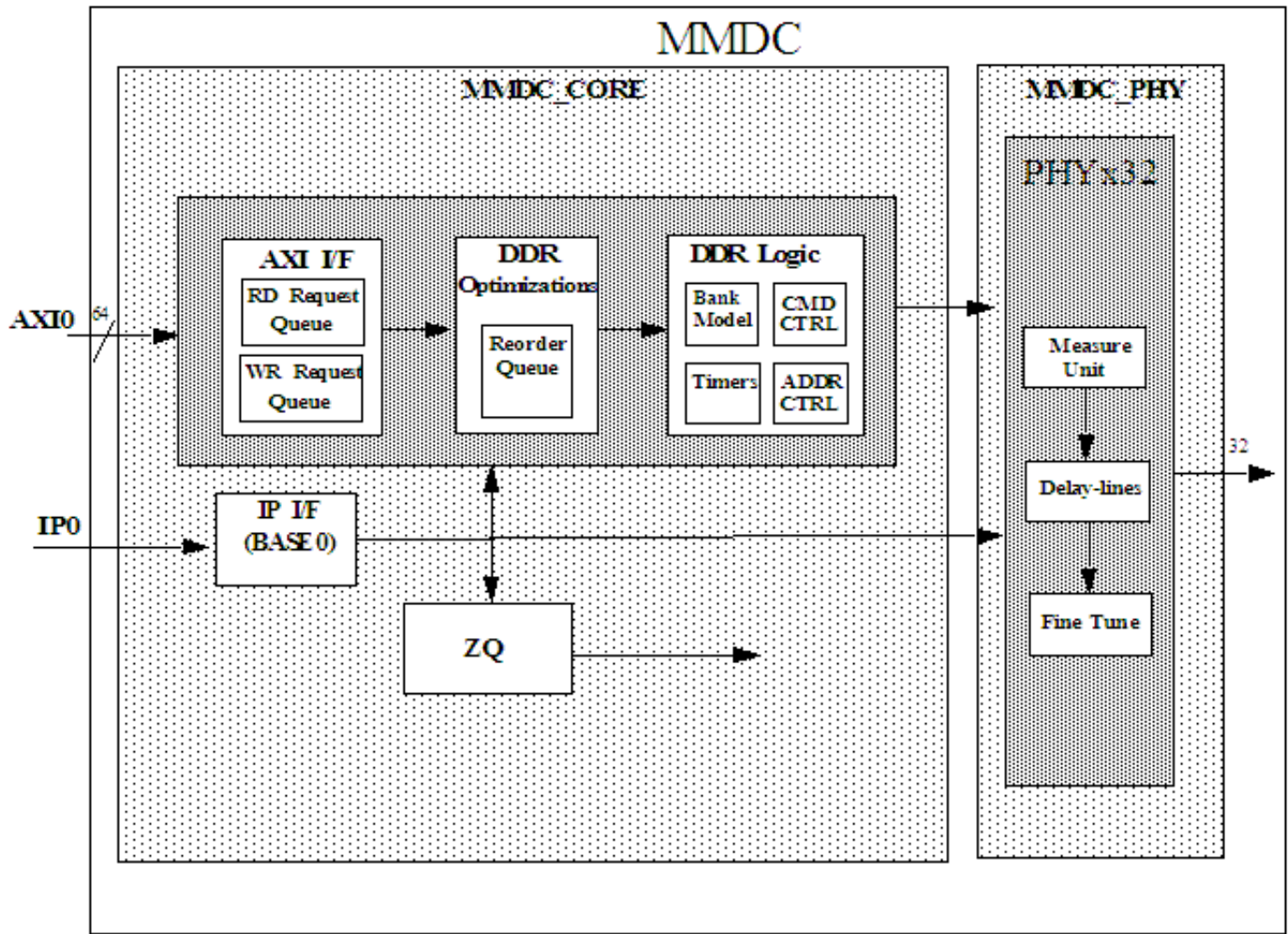
## Chapter 32

# Multi Mode DDR Controller (MMDC)

### 32.1 Overview

MMDC is a multi-mode DDR controller that supports DDR3/DDR3L x16/x32 and LPDDR2 x16/x32 memory types. MMDC is configurable, high performance, and optimized.

The following figure shows the MMDC block diagram.



**Figure 32-1. MMDC block diagram**

MMDC consists of a core (MMDC\_CORE) and PHY (MMDC\_PHY).

- The core is responsible for communication with the system through an AXI interface, DDR command generation, DDR command optimizations, and a read/write data path.
- The PHY is responsible for the timing adjustment; it uses special calibration mechanisms to ensure data capture margin at a clock rate of up to 400 MHz.

The internal memory map(configuration registers) of the MMDC can be configured through an IP channel (IPO). .



### 32.1.1 MMDC feature summary

The table found here summarizes the MMDC features.

**Table 32-1. MMDC feature summary**

Feature	Details
DDR standards	<ul style="list-style-type: none"> <li>• LV-DDR3, DDR3 x16, x32</li> <li>• LPDDR2 x16, x32</li> <li>• Does not support LPDDR1MDDR or DDR2</li> </ul>
DDR interface	<ul style="list-style-type: none"> <li>• x16, x32 data bus width</li> <li>• Density per DDR device of 256 Mbits–8 Gbits with the following column and row combinations: <ul style="list-style-type: none"> <li>• Column size of 8–12 bits</li> <li>• Row size of 11–16 bits</li> </ul> </li> <li>• Two chip selects</li> <li>• Up to 4 Gbytes of address space with configurable partitioning between CS0 and CS1</li> <li>• Supports burst length of 8 (aligned) for DDR3</li> <li>• Supports burst length of 4 for LPDDR2</li> </ul>
DDR performance	<ul style="list-style-type: none"> <li>• MMDC running at up to 400 MHz (800MT/s), see CCM block for actual clock frequencies supported.</li> <li>• Supports Real-Time priority by means of QoS sideband priority signals from the chip to enable various priority levels in the re-ordering mechanism: real-time, latency sensitive, normal priority.</li> <li>• Page hit/page miss optimizations</li> <li>• Consecutive read/write access optimizations</li> <li>• Supports deep read and write request queues to enable bank prediction.</li> <li>• Drives back the critical word in a read transaction as soon as it is received by the DDR device (does not wait until the whole data phase has been completed).</li> <li>• Keeps tracking of open memory pages</li> <li>• Supports bank interleaving</li> <li>• Special optimization in case of non-aligned wrap accesses in DDR3 mode (burst length 8)</li> </ul> <p><b>NOTE:</b> Due to reordering and optimization mechanisms (per different AXI Identifier (ID)), the transactions towards the DDR device may be driven in a different ID order than was received by the AXI master. In a similar fashion, the write response, read response or read data may be driven to the AXI master in a different ID order.</p>

*Table continues on the next page...*

**Table 32-1. MMDC feature summary (continued)**

Feature	Details
AXI interface	<ul style="list-style-type: none"> <li>• AXI bus compliant</li> <li>• Supports bus transfers of 8, 16, 32, 64 bits (single accesses and bursts) running at 400 MHz.</li> <li>• Supports AXI bursts length of up to 16</li> <li>• Supports burst types of WRAP, INCR and FIXED</li> <li>• Supports 16 bits AXI ID</li> <li>• Write data interleave depth is 1 (no support for Write Data Interleave)</li> <li>• Supports write data before address</li> <li>• Supports buffered/non-buffered accesses (AWCACHE[0] = 0b means a non-bufferable access and AWCACHE[0] = 1b means a bufferable access). The rest of the CACHE options are not supported               <ul style="list-style-type: none"> <li>• To keep data access coherency between write and read access of the same master, the response signal is sent as follows:                   <ul style="list-style-type: none"> <li>• Bufferable write access—BRESP will be sent when last data of the access has entered the MMDC.</li> <li>• Non-bufferable write access—BRESP will be sent when the data was physically written into the external memory device.</li> </ul> </li> </ul> </li> <li>• Supports four exclusive monitors per configurable ID for only a single access with a size of up to 64 bits</li> <li>• Supports AXI responses as follows:               <ul style="list-style-type: none"> <li>• Okay in case the access has been successful or exclusive access failure</li> <li>• Slave error in case of security violation</li> <li>• Exclusive okay in case the read or the write portion of an exclusive access has been successful</li> </ul> </li> </ul>
DDR calibration and delay-lines.	<ul style="list-style-type: none"> <li>• Supports various calibration processes which can be performed either automatically (hardware) or manually (software) towards either CS0 or CS1. (At the end of the process the delay-lines will work with one set of results.) The following calibration processes are supported:               <ul style="list-style-type: none"> <li>• ZQ calibration for external DDR device (in DDR3 through ZQ calibration command and in LPDDR2 through MRW command)                   <ul style="list-style-type: none"> <li>• Can be handled automatically for ZQ Short (periodically) and ZQ Long (at exit from self-refresh)</li> <li>• Can be handled manually at ZQ INIT</li> </ul> </li> <li>• ZQ calibration for i.MX DDR I/O pads for calibrating the DDR driving strength                   <ul style="list-style-type: none"> <li>• The sequence can be handled automatically by hardware</li> <li>• The sequence can be handled step by step manually by software</li> </ul> </li> <li>• Read data calibration. Adjustment of read DQS with read data byte.</li> <li>• Read DQS gating calibration for DDR3 only. Adjustment of DQS gate with read preamble window.</li> <li>• Write data calibration. Adjustment of write DQS with write data byte.</li> <li>• Write leveling calibration. Adjustment of write DQS with CK (DDR differential clock).</li> <li>• Read fine tuning. Adjustment of up to 7 delay-line units for each read data bit.</li> <li>• Write fine tuning. Adjustment of up to 3 delay-line units for each read data bit.</li> <li>• Periodic delay-line measurement for keeping its accuracy during refresh interval.</li> <li>• Additional fine tuning delay lines to adjust DDR clock delay, DDR clock duty cycle, DQS duty cycle.</li> </ul> </li> </ul>

*Table continues on the next page...*

**Table 32-1. MMDC feature summary (continued)**

Feature	Details
Power saving	<ul style="list-style-type: none"> <li>Support of dynamic voltage, frequency change and self-refresh mode entry through hardware and software negotiation with the system (request/acknowledge handshake) <ul style="list-style-type: none"> <li>Upon hardware or software self-refresh request assertion, further AXI requests are blocked (even before the assertion of the acknowledge).</li> <li>During self-refresh mode the system may deassert the operating clock of the MMDC for power saving.</li> <li>During self-refresh mode the clock (CK) that is driven to the DDR device will be gated for power saving.</li> </ul> </li> <li>Supports automatic self-refresh and power down entry and exit <ul style="list-style-type: none"> <li>In automatic self-refresh, the internal operating clock will be gated for power saving.</li> </ul> </li> <li>Supports fast and slow precharge power down in DDR3</li> <li>Automatic active and precharge power down timer per chip select (one chip select can enter power down while the other is still working)</li> <li>While CS (chip-select) is inactive (high) the command and address buses are not toggling for power saving.</li> <li>While DM (data masking) is high the associated DQ bus is not toggling (driven to "0") for power saving.</li> </ul>
DDR general	<ul style="list-style-type: none"> <li>Configurable timing parameters</li> <li>Configurable refresh scheme</li> <li>Page boundary crossing support <ul style="list-style-type: none"> <li>Automatically generates precharge command and activates the next row</li> </ul> </li> <li>Supports various ODT control schemes <ul style="list-style-type: none"> <li>Assertion or deassertion of ODT control per read or write accesses and for active or passive CS (chip-select)</li> </ul> </li> <li>Supports MRW and MRR commands for LPDDR2</li> <li>Software control in LPDDR2 mode for switching to derated timing parameters and/or update the refresh rate according to temperature sensor</li> <li>Debug and profiling capabilities</li> </ul>

## 32.2 External Signals

The table found here describes the external signals of MMDC.

**Table 32-2. MMDC External Signals**

Signal	Description	Pad	Mode	Direction
DRAM_ADDR[15:00]	Address Bus Signals	DRAM_A[15:0]	No Muxing	I/O
DRAM_CAS	Column Address Strobe Signal	DRAM_CAS	No Muxing	I/O
DRAM_CS[1:0]	Chip Selects	DRAM_CS[1:0]	No Muxing	I/O
DRAM_DATA[31:00]	Data Bus Signals	DRAM_D[31:0]	No Muxing	I/O
DRAM_DQM[7:0]	Data Mask Signals	DRAM_DQM[7:0]	No Muxing	I/O
DRAM_ODT[1:0]	On-Die Termination Signals	DRAM_SDODT[1:0]	No Muxing	I/O
DRAM_RAS	Row Address Strobe Signal	DRAM_RAS	No Muxing	I/O
DRAM_RESET	Reset Signal	DRAM_RESET	No Muxing	I/O
DRAM_SDBA[2:0]	Bank Select Signals	DRAM_SDBA[2:0]	No Muxing	I/O

*Table continues on the next page...*

**Table 32-2. MMDC External Signals (continued)**

Signal	Description	Pad	Mode	Direction
DRAM_SDCKE[1:0]	Clock Enable Signals	DRAM_SDCKE[1:0]	No Muxing	I/O
DRAM_SDCLK0_N	Negative Clock Signal 0	DRAM_SDCLK_0_B	No Muxing	I/O
DRAM_SDCLK0_P	Positive Clock Signal 0	DRAM_SDCLK_0	No Muxing	I/O
DRAM_SDWE	WE signal	DRAM_SDWE	No Muxing	I/O

## 32.3 Clocks

The table found here describes the clock sources for MMDC.

Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 32-3. MMDC Clocks**

Clock name	Clock Root	Description
aclk_fast_core_p0	mmdc_axi_clk_root	Fast clock (channel 1)
ipg_clk_p0	ipg_clk_root	Peripheral clock (channel 1)
aclk_fast_phy_p0	mmdc_axi_clk_root	Fast clock (channel 1 - PHY)

## 32.4 Functional Description

This section provides a complete functional description of the block.

### 32.4.1 Write/Read data flow

#### 32.4.1.1 Write data flow

- Write requests are received into an 8 entries request FIFO. Access is received only when there are at least two available entries. Each entry holds all of the AXI attributes.
  - If the burst length is greater than 8, the access splits into two accesses: one with burst length 8 and the other with the remainder.
  - The access can be performed as soon as the entire data phase of the associated write request is completed (all data beats were received).

2. A simple round-robin arbitration between the pending read and write accesses is performed, and the pointer to this stage's winner access is sent to the re-ordering buffer.
3. The reordering mechanism is activated to find the winner access, which is the access that best utilizes the DDR bus, based on its dynamic score. For further information see [Dynamic scoring mode \(Arbitration Winning Conditions\)](#).
4. The winner write access at the previous stage is received and is held for dispatch to the DDR logic.
5. When the DDR command control unit is ready to accept the write request, it issues (if needed) a precharge/active command to the DDR device according to the status of the bank model and the parameters of the timers.
6. The DDR logic drives the associated data to the DDR device through the DDR PHY.

### 32.4.1.2 Read data flow

1. Read requests are received into a 16 entry request FIFO in MMDC if there are at least two available entries. Each entry holds all of the AXI attributes.

#### NOTE

If the burst length is greater than 8, the access splits into 2 accesses (one with burst length 8 and the other with the remainder).

2. A simple round-robin arbitration between the pending read and write accesses is performed and the pointer to this phase's winner access is sent to the re-ordering buffer.
3. The reordering mechanism is activated to find the winner access, which is the access that best utilizes the DDR bus, based on its dynamic score. For further information see [Dynamic scoring mode \(Arbitration Winning Conditions\)](#).
4. The winner read access at the previous stage is sampled and is held for dispatch to the DDR logic. This read access will be dispatched when there is at least one free slot in the read data buffer to store the data.
5. When the DDR command control unit is ready to accept the read request, it issues (if needed) a precharge/active command to the DDR device according to the status of the bank model and the parameters of the timers.
6. The MMDC PHY samples the read data, and the DDR logic transfers the data to the associated slot in the read data buffer.
7. MMDC transfers the data back to the master.

## 32.4.2 MMDC initialization

Because the MMDC is disabled when the chip exits reset, no clock is driven to the DDR device and the whole interface towards the DDR device is inactive. The following steps are required to activate the MMDC properly.

### NOTE

To guarantee that the DRAM\_RESET and DRAM\_SDCKE signals are kept low during the power-up and reset sequences of the chip in DDR3 and LPDDR2 modes (as defined by JEDEC), you must connect those signals to pull-down resistors.

1. Set MDSCR[CON\_REQ], which sets the configuration request; note that because the MMDC is disabled, there is no need to poll the configuration acknowledge bit at MDSCR[CON\_ACK].
2. Configure the desired timing parameters at the MDCFG0, MDCFG1, MDCFG2, and MDOTC registers.
3. Configure the DDR type and other miscellaneous parameters at the MDMISC register.
4. Configure the required delay while leaving reset, at the MDOR register.
5. Configure the DDR physical parameters (density and burst length) at the MDCTL register.
6. Perform a ZQ calibration of the MMDC module to correctly initialize drive strengths.
7. Enable MMDC with the desired chip select at MDCTL[SDE\_0] (for chip select 0) and MDCTL[SDE\_1] (for chip select 1). At this point, MMDC starts the reset and initialization sequence related to DRAM\_RESET/DRAM\_SDCKE as defined by JEDEC.
8. Complete the initialization sequence as defined by JEDEC by issuing MRS/MRW commands for (ZQ, ODT, PRE, and so on). To issue those commands, configure the appropriate command and address at the MDSCR register.
9. Program the DDR mode registers by configuring the appropriate command and address at the MDSCR register.
10. Configure the power down and self-refresh entry and exit parameters at the MDPDC and MAPSR registers.
11. Configure the ZQ scheme at the MPZQHWCTRL and MPZQLP2CTRL registers.
12. Configure and activate the periodic refresh scheme at the MDREF register.
13. Deassert the configuration request by clearing MDSCR[CON\_REQ].

### NOTE

Steps 1 through 5 are non-blocking and can be done in any order.

Upon completion of these steps, MMDC is ready for work and to process AXI accesses.

**NOTE**

To achieve better timing and better precision, it is recommended that users configure the MMDC PHY delay parameters by operating either the automatic or manual calibration process. Before starting any calibration process, you must disable the periodic refresh scheme (MDREF[REF\_SEL] = 00) and then issue a manual refresh command by configuring MDSCR[CMD] to 2h. For further information, see [Calibration Process](#).

### 32.4.3 Configuring the MMDC registers

To safely modify MMDC's internal configuration registers, MMDC must be placed into configuration mode.

Use the following steps to enter configuration mode.

1. Issue a configuration request by setting MDSCR[CON\_REQ].
2. Poll on configuration acknowledge until it is set at MDSCR[CON\_ACK].

At this point, MMDC enters configuration mode and accessing the MMDC registers is permitted.

**NOTE**

During configuration mode, MMDC prevents further AXI accesses from being acknowledged.

Upon deassertion of MDSCR[CON\_REQ], MMDC leaves configuration mode and AXI accesses are processed.

### 32.4.4 MMDC Address Space

#### 32.4.4.1 Address decoding

MMDC supports up to two consecutive chip selects, each with the same density.

It is optional to configure the partition between the chip selects through MDASP[CS0\_END].

The incoming AXI address bus is 32 bits. MMDC decodes each access as follows:

1. chip select

2. bank number
3. row number
4. column number

The following registers in the MMDC define the DDR address space:

- MDMISC[DDR\_4\_BANK]—Defines either 4 or 8 banks in the DDR device
- MDCTL[DSIZ]—Defines the DDR data bus width of x16, x32 or x64
- MDMISC[BI]—Defines whether bank interleaving is on or off
- MDCTL[COL]—Defines the column size of the DDR device
- MDCTL[ROW]—Defines the row size of the DDR device

The following tables show address decoding examples for x16 and x32 bit DDR devices when bank interleaving is both on and off. It is assumed that the configuration is as follows: 8 banks (3 bits), 15 bit assignment for the row, and 10 bit assignment for the column. The total density is 256 MWords (512 Mbytes for x16 and 1 Gbyte for x32).

### NOTE

Chip selection is done by comparing the 7 most significant address bits (ARADDR[31:25]/AWADDR[31:25]) with MDASP[CS0\_END].

**Table 32-4. Address decoding—bank interleaving off**

AXI ADDRESS	x16 DDR	x32 DDR
A29	—	BANK[2]
A28	BANK[2]	BANK[1]
A27	BANK[1]	BANK[0]
A26	BANK[0]	ROW[14]
A25	ROW[14]	ROW[13]
A24	ROW[13]	ROW[12]
A23	ROW[12]	ROW[11]
A22	ROW[11]	ROW[10]
A21	ROW[10]	ROW[9]
A20	ROW[9]	ROW[8]
A19	ROW[8]	ROW[7]
A18	ROW[7]	ROW[6]
A17	ROW[6]	ROW[5]
A16	ROW[5]	ROW[4]
A15	ROW[4]	ROW[3]
A14	ROW[3]	ROW[2]
A13	ROW[2]	ROW[1]
A12	ROW[1]	ROW[0]
A11	ROW[0]	COL[9]

*Table continues on the next page...*



**Table 32-4. Address decoding—bank interleaving off (continued)**

AXI ADDRESS	x16 DDR	x32 DDR
A10	COL[9]	COL[8]
A9	COL[8]	COL[7]
A8	COL[7]	COL[6]
A7	COL[6]	COL[5]
A6	COL[5]	COL[4]
A5	COL[4]	COL[3]
A4	COL[3]	COL[2]
A3	COL[2]	COL[1]
A2	COL[1]	COL[0]
A1	COL[0]	—
A0	—	—

**Table 32-5. Address decoding—bank interleaving on**

AXI ADDRESS	x16 DDR	x32 DDR
A29	—	ROW[14]
A28	ROW[14]	ROW[13]
A27	ROW[13]	ROW[12]
A26	ROW[12]	ROW[11]
A25	ROW[11]	ROW[10]
A24	ROW[10]	ROW[9]
A23	ROW[9]	ROW[8]
A22	ROW[8]	ROW[7]
A21	ROW[7]	ROW[6]
A20	ROW[6]	ROW[5]
A19	ROW[5]	ROW[4]
A18	ROW[4]	ROW[3]
A17	ROW[3]	ROW[2]
A16	ROW[2]	ROW[1]
A15	ROW[1]	ROW[0]
A14	ROW[0]	BANK[2]
A13	BANK[2]	BANK[1]
A12	BANK[1]	BANK[0]
A11	BANK[0]	COL[9]
A10	COL[9]	COL[8]
A9	COL[8]	COL[7]
A8	COL[7]	COL[6]
A7	COL[6]	COL[5]
A6	COL[5]	COL[4]

*Table continues on the next page...*

**Table 32-5. Address decoding—bank interleaving on (continued)**

AXI ADDRESS	x16 DDR	x32 DDR
A5	COL[4]	COL[3]
A4	COL[3]	COL[2]
A3	COL[2]	COL[1]
A2	COL[1]	COL[0]
A1	COL[0]	—
A0	—	—

**NOTE**

In cases where this is an access to a non-initialized or disconnected chip select, behavior may be unexpected.

**32.4.4.2 Chip select settings**

MMDC drives the incoming access to either CS0 or CS1 by comparing the 7 most significant address bits (ARADDR[31:25]/AWADDR[31:25]) with MDASP[CS0\_END].

Generally, the total density per chip-select must be the same, and the total density per chip-select must be a power of two.

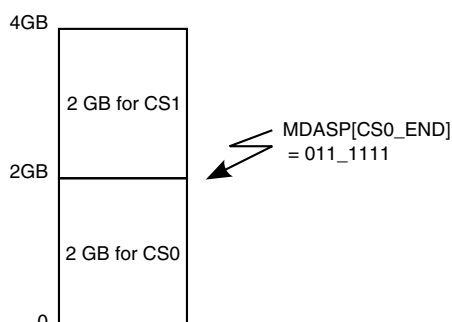
[Creating 4 Gbyte address space with 2 Gbyte CS density](#) and [Creating 2 Gbyte address spaces with 1 Gbyte CS density](#) show how to create a continuous address space and configure the MMDC accordingly.

**32.4.4.2.1 Creating 4 Gbyte address space with 2 Gbyte CS density**

If the DDR memory space allocation is 4 Gbytes, only one configuration of chip select partition is allowed.

The register MDASP[CS0\_END] should be set to 011\_1111 (partition at 2 Gbytes).

The figure below shows the associated memory space. In the case of DDR3 x64, this address space can be achieved by connecting four devices per chip select. Each device is x16 with density of 4 Gbytes.



**Figure 32-2. Chip select partition—2 Gbytes per chip select**

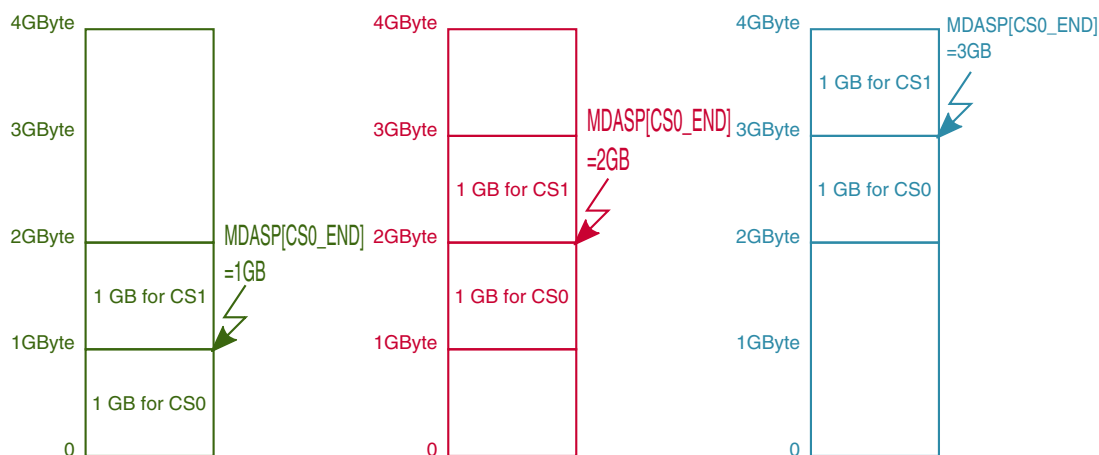
### 32.4.4.2.2 Creating 2 Gbyte address spaces with 1 Gbyte CS density

If the DDR memory space allocation is 2 Gbytes, there are three options for configuring the chip select partition: MDASP[CS0\_END] to 001\_1111 (1 Gbyte), MDASP[CS0\_END] to 011\_1111 (2 Gbytes), and MDASP[CS0\_END] to 101\_1111 (3 Gbytes).

If DDR memory space allocation is 2 Gbytes, there are three options for configuring the chip select partition:

- MDASP[CS0\_END] to 001\_1111 (1 Gbyte)
- MDASP[CS0\_END] to 011\_1111 (2 Gbytes)
- MDASP[CS0\_END] to 101\_1111 (3 Gbytes)

The figure below shows the associated memory space:



**Figure 32-3. Chip select partition—1 Gbyte per chip select**

### 32.4.4.3 Translation of AXI accesses to DDR accessesss

### 32.4.4.3.1 Example

Assume the AXI write access has the following attributes:

- Increment (awburst[1:0]=2'b01)
- AXI size of 64bits (awsize[2:0]=3'b011)
- AXI length of 2 (awlen[3:0]=4'b0001)
- AXI address with suffix 0x5 (non aligned)

Toward DDR3(MDMISC[DDR\_TYPE]=2'b00) with the following attributes:

- x32 (MDCTL[DSIZ]=2'b10)
- burst length of 8 (MDCTL[BL]=1'b1)

In this case the AXI alignment is every 8B (0x8) and the DDR boundary is every 4Bx8=32B(0x20).

The master expects to write the data to the following addresses: 0x5 (with WSTRB=0xE0), 0x8 (till 0xF).

The MMDC will issue one access toward the DDR as follows:

Write access toward logic address with suffix 0x0 (DDR boundary is 0x20 and 0x0 is the closest to 0x0) while address 0x0 till 0x4 are masked by DM (data masking signal) and address 0x10 till 0x1F are also masked by DM.

### 32.4.4.4 Address mirroring

When enabling this feature, address bits DRAM\_A3, DRAM\_A4, DRAM\_A5, DRAM\_A6, DRAM\_A7, DRAM\_A8, DRAM\_SDBA0, and DRAM\_SDBA1 behave differently according to the associated chip select.

This feature facilitates PCB board routing for devices on chip select 1, which are typically populated on the opposite side of the PCB from the devices on chip select 0.

#### NOTE

This feature is only supported for DDR3 memories. It is not supported for LPDDR2 memories.

The following table specifies the address mirroring options:

**Table 32-6. Address mirroring options**

MMDC pin	Chip select 0 pin	Chip select 1 pin
DRAM_A3	DRAM_A3	DRAM_A4

*Table continues on the next page...*

**Table 32-6. Address mirroring options (continued)**

MMDC pin	Chip select 0 pin	Chip select 1 pin
DRAM_A4	DRAM_A4	DRAM_A3
DRAM_A5	DRAM_A5	DRAM_A6
DRAM_A6	DRAM_A6	DRAM_A5
DRAM_A7	DRAM_A7	DRAM_A8
DRAM_A8	DRAM_A8	DRAM_A7
DRAM_SDBA0	DRAM_SDBA0	DRAM_SDBA1
DRAM_SDBA1	DRAM_SDBA1	DRAM_SDBA0

### 32.4.5 LPDDR2 and DDR3 pin mux mapping

The following table shows the pin mux mapping between LPDDR2 and DDR3. The i.MX DDR I/O pads corresponds with the DDR3 standard.

- In DDR3, all DRAM\_DATA, DRAM\_SDQS, and DRAM\_DQM data lines work with channel 0.
- In LPDDR2, DRAM\_DDQS[3:0], DRAM\_DATA[31:0] and DRAM\_DQM[3:0] work with channel 0. DRAM\_SDQS[7:4], DRAM\_DATA[63:32], and DRAM\_DQM[7:4] work with channel 1.

**Table 32-7. LPDDR2 and DRAM pin mux mapping**

DRAM I/O pad	LPDDR2 I/O pad
DRAM_ADDR00	LPDDR2_CA0
DRAM_ADDR01	LPDDR2_CA1
DRAM_ADDR02	LPDDR2_CA2
DRAM_ADDR03	LPDDR2_CA3
DRAM_ADDR04	LPDDR2_CA4
DRAM_ADDR05	LPDDR2_CA5
DRAM_ADDR06	LPDDR2_CA6
DRAM_ADDR07	LPDDR2_CA7
DRAM_ADDR08	LPDDR2_CA8
DRAM_ADDR09	LPDDR2_CA9
DRAM_ADDR10	—
DRAM_ADDR11	—
DRAM_ADDR12	—
DRAM_ADDR13	—
DRAM_ADDR14	—
DRAM_ADDR15	—

*Table continues on the next page...*

**Table 32-7. LPDDR2 and DRAM pin mux mapping (continued)**

DRAM I/O pad	LPDDR2 I/O pad
DRAM_CAS_B	—
DRAM_RAS_B	—
DRAM_WE_B	—
DRAM_SDCKE0	LPDDR2_CKE0
DRAM_SDCKE1	LPDDR2_CKE1
DRAM_CS_B0	LPDDR2_CS_B0
DRAM_CS_B1	LPDDR2_CS_B1
DRAM_ODT0	LPDDR2_ODT0
DRAM_ODT1	LPDDR2_ODT1
DRAM_SDCLK0_P	LPDDR2_CK0
DRAM_SDCLK1	LPDDR2_CK1
DRAM_BA0	—
DRAM_BA1	—
DRAM_BA2	—

## 32.4.6 Power Saving and Clock Frequency Change modes

### 32.4.6.1 Power saving general

MMDC supports multiple DDR power saving modes.

#### NOTE

At default, the power saving modes are disabled. These modes may dramatically decrease the power consumption of DDR memories.

- Self-refresh entry to the entire DDR device (for both chip select 0 and 1) can be activated through two mechanisms:
  - LPMD (Low Power Mode)
    - Hardware handshaking (LPMD/LPACK) with the clock module in the system
    - Software handshaking by setting the field MAPSR[LPMD] and polling MAPSR[LPACK]
    - Automatic entry by configuring the amount of idle cycle for triggering self-refresh entry through MAPSR[PST] and by clearing MAPSR[PSD]
  - DVFS (Dynamic Voltage and Frequency Change)

- Hardware handshaking (DVFS/DVACK) with the clock module in the system
- Software handshaking by setting the field MAPSR[DVFS] and polling MAPSR[DVACK]

### NOTE

If hardware or software requests for self-refresh entry were detected by the MMDC (even before the assertion of the LPACK), no write or read accesses will be acknowledged until the deassertion of those requests.

2. Automatic active/precharge power down entry to a specific chip select can be activated by configuring the ESDPDC register:
  - PWDT\_0/PWDT\_1 - define the number of idle cycles before entering power down, can be different value per chip select.
  - SLOW\_PD - In case of DDR3 memory is configured to use slow precharge power down then this bit should be set as well.
  - BOTH\_CS\_PS - The MMDC can either set each chip select independently to power down, according to its idle state, or set both chip selects to power down only if both in idle state for the configured period.
  - Few parameters must be configured in addition:
    - Timing parameters at ESDCFG0[tXP and tXPDLL].
    - ODT timing at ESDOTC[tAOFPD, tAONPD, tANPD and tAXPD]

### NOTE

It is possible to enter certain chip selects to low power consumption while the second chip select is activated.

3. Automatic precharge of all DDR banks to a specific chip select. Can be activated by configuring ESDPDC fields: PRCT\_0 and PRCT\_1. Each field determines a value loaded to a different chip select.

#### 32.4.6.2 Self refresh and Frequency change entry/exit

As described in [Power saving general](#), the MMDC supports two mechanisms that will cause the DDR device to enter self-refresh mode:

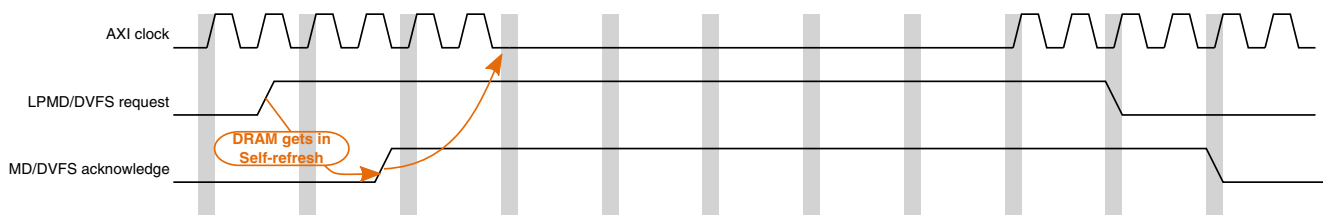
- LPMD (Low Power Mode) - For power saving purposes
- DVFS (Dynamic Voltage and Frequency Change) - For clock frequency changes

While the DDR device is in self-refresh mode, there is no need to provide periodic refresh commands.

The MMDC treats hardware/software handshaking of LPMD/DVFS in the same manner:

- Upon the assertion of LPMD/DVFS request, the following is done:
  - The MMDC blocks any further AXI accesses even before the acknowledge is asserted
  - Completes all opened AXI accesses
  - Closes (precharge) all banks in the appropriate timing
  - Drives self-refresh command by deasserting clock enable signal (DRAM\_SDCKE is driven to "0") together with a refresh command. This occurs after satisfying tRP/tRPA from the precharge all command.
  - Deasserts the clock (CK) that is driven to the DDR device
  - Asserts LPMD/DVFS acknowledge (LPACK/DVACK)
  - Allows deassertion of the operating clock of the MMDC (AXI clock)
- Upon the deassertion of LPMD/DVFS request, the following is done:
  - Operating clock of the MMDC must be turned on before LPMD/DVFS is deasserted
  - Starts driving the clock (CK) to the DDR device
  - After satisfying tCKSRX from clock renewal the clock enable signal (DRAM\_SDCKE) is asserted
  - LPMD/DVFS acknowledge (LPACK/DVACK) is deasserted
  - After satisfying tXS from the assertion of DRAM\_SDCKE, a refresh command is driven to the DDR device.
  - If ZQ calibration is enabled then tRFC is satisfied from the refresh command and a long ZQ command is driven.
  - tZQoper idle cycles are counted after the ZQ command.
  - After satisfying tDLLK from the assertion DRAM\_SDCKE, the MMDC returns to normal operation.

The figure below shows the timing diagram of the hardware/software handshaking of LPMD/DVFS:



**Figure 32-4. LPMD/DVFS Hardware/Software Handshaking**

Note for self-refresh:



- As soon as LPMD or DVFS requests are detected by either hardware or software handshaking, the MMDC will deassert the AXI ARREADY/AWREADY signals immediately to block further requests from the system.
- In case of automatic self-refresh, the internal operating clock will be negated to save power.

## 32.4.7 Reset

### 32.4.7.1 Hard reset

When hard reset is asserted (aresetn is driven to "0") while warm reset is deasserted (warm\_reset is driven to "0"), the entire MMDC will be initialized, including configuration/status registers and state machines.

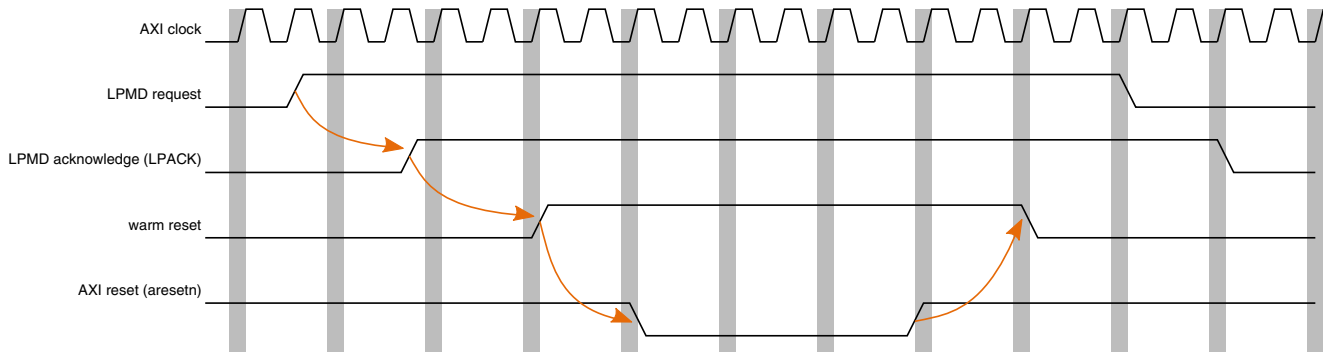
In order to access the DDR device, the MMDC will then have to be reconfigured.

### 32.4.7.2 Warm reset

The MMDC supports warm reset signal. The warm reset signal must envelop the hard reset signal and then the MMDC will reset all the internal registers. The only registers that are not reset are those that are essential for returning it to normal operation without repeating the initialization sequence and without losing data stored in the memory (configuration/status registers won't be initialized).

For the successful operation of warm reset, the following steps must be performed:

- The MMDC must enter self-refresh mode. This can be achieved by either LPMD or DFVS requests
- Wait for LPMD or DVFS acknowledge
- Assert warm reset signal (i.e. drive warm\_reset to "1")
- Assert hard reset signal (i.e. drive aresetn to "0")
- Deassert hard reset signal
- Deassert warm reset
- Get out of the LPMD/DVFS mode



**Figure 32-5. Warm Reset Diagram**

### 32.4.7.3 Software reset

The MMDC supports software reset. When software reset is configured then the MMDC will reset all the internal registers except those that are essential for returning to normal operation without repeating the initialization sequence or without losing data stored in the memory (configuration/status registers won't be initialized).

The following steps should be performed for successful operation of software reset:

- The MMDC should enter self-refresh mode. This can be achieved by either LPMD or DFVS request.
- Wait for LPMD or DVFS acknowledge
- Assert software reset, by setting MDMISC[RST]
- Get out of the LPMD/DVFS mode

Normal operation can be resumed.

### 32.4.8 Refresh Scheme

The MMDC supports various automatic refresh options which can be configured via the MDREF register.

The periodic auto refresh can be triggered by the following clocks:

- 32KHz clock
- 64KHz clock
- MMDC operating clock

The refresh scheme of the MMDC is flexible and allows the system to configure the desired AXI accesses delay/latency in each refresh cycle.

The table below shows an example of four configurations of the refresh cycles that will be handled by the MMDC. Each configuration meets a refresh rate of 3.9us (tREFI, refresh command every 3.9us).

**Table 32-8. MMDC Refresh Scheme**

Option number	Description	REFR	REF_SEL	REF_CNT	DDR hang time
1	Issue 8 refresh commands every 31,250 ns	0x7 (8 refreshes)	0x2 (64KHz)	not needed	tRFC * 8
2	Issue 4 refresh commands every 15,625ns	0x3 (4 refreshes)	0x1(32KHz)	not needed	tRFC * 4
3	Issue 2 refresh commands every 7800ns	0x1(2 refreshes)	0x3 (fast counter)	7800/2.5 = 3120 (0xC30)	tRFC * 2
4	Issue 1 refresh command every 3900 ns	0x0 (1 refresh)	0x3 (fast counter)	3900/2.5 = 1560(0x618)	tRFC

### 32.4.9 Burst Length options towards DDR

The MMDC supports two kinds of burst lengths which can be configured through MDCTL[BL] as follows:

- In DDR3 mode, only burst length 8 can be used.
- In LPDDR2 mode, only burst length 4 can be used.

In DDR3 mode read/write accesses to the DDR are always 8 words (x16, x32, x64) and aligned in according to JEDEC standards.

In case of AXI INCREMENT, accesses that are not aligned the irrelevant data is masked in write accesses and ignored in read accesses. In case of AXI WRAP accesses, even if the access is not aligned, then the MMDC provides an internal optimization mechanism for better efficiency of the DDR data bus.

### 32.4.10 Exclusive accesses handling

The MMDC contains four exclusive monitors, each for dedicated ID as configured in MAEXIDR0 and MAEXIDR1.

- If legal read exclusive is received by the MMDC, the associated monitor is turned on.

- While the monitor is turned on upon legal write exclusive, the monitor will be turned off and the write will be completed successfully with EXOKAY.
- The following rules must be met for successful exclusive access:
  - Aligned access (the AXI address is aligned to the AXI size)
  - AXI single access (AXI burst length isn't greater than 1)
  - AXI size of up to 64 bits
  - AXI non-cachable access (i.e. ARCACHE[1]/AWCACHE[1] is equal "0" or ARCACHE[1]/AWCACHE[1] is equal "1" while ARCACHE[3:2]/AWCACHE[3:2] are equal "00")
  - AXI ID that matches one of the four exclusive IDs

Exclusive read behavior (first bullet also correct for non-exclusive accesses):

- In case of security violation, the read is blocked and is not sent to DDR. There are two options for response:
  - If ARCR\_SEC\_ERR\_EN (MAARCR[30]) is high, SLV error is issued towards the Master, otherwise OKAY response is sent to the Master.
- If AXI exclusive rules violation occurs (as described above), the read access is not blocked and is sent to DDR. The data will be fetched and be driven to the master, but the type of response may be unpredicted.
- If none of the above occurs, the read is sent to the DDR. The exclusive monitor will be turned on and the response is ExOKAY
- If additional legal AXI read exclusive is received with the same ID before the AXI exclusive write, the monitor will be updated with the latest attributes.

Exclusive write behavior (first bullet also correct for non-exclusive accesses):

- In case of security violation, the write is blocked and is not sent to DDR, but the monitor will be kept on. There are two options for response:
  - If ARCR\_SEC\_ERR\_EN (MAARCR[30]) is high then SLV error is issued towards the Master, otherwise OKAY response is sent to the Master.
- In case of AXI exclusive rules violation (as described above), the write is blocked and is not sent to DDR. In that case the type of response may be unpredicted.
- In case the exclusive write access has different AXI attributes, but the same ID as the read exclusive access, the write is blocked and is not sent to DDR and the monitor will be turned off. There are two options for response:
  - If ARCR\_EXC\_ERR\_EN (MAARCR[28]) is high then SLV error is issued towards the Master, otherwise OKAY response is sent to the Master.
- In case of regular (non exclusive) write access is received to the same address or overlapping addresses then the write will be sent to the DDR and the monitor will be turned off.
- In case of legal write exclusive access is received with the same attributes as the read exclusive access while the monitor is on ( no write accesses occurred to the same

address between the read exclusive and write exclusive), then the write is sent to DDR and the response is EXOKAY. But, if the legal write exclusive is received while the monitor is off, the write is blocked and there are two options for response.

- If ARCR\_EXC\_ERR\_EN (MAARCR[28]) is high then SLV error is issued towards the Master, otherwise OKAY response is sent to the Master.

### 32.4.11 AXI Error Handling

The MMDC supports the AXI responses listed here.

- In case of AXI exclusive violation there are two options for response:
  - If MAARCR[28] is high then SLVError is issued towards the Master, Otherwise OKAY response is sent to the Master

#### NOTE

In case of read error MMDC drives zeros on the read data bus

## 32.5 Performance

### 32.5.1 Arbitration and reordering mechanism

#### 32.5.1.1 Arbitration General

The following specifies arbitration and reordering flow in MMDC towards the DDR.

- AXI read and write accesses are sampled in the associated queue.
- Read/write arbitration is handled to select the winning access.
- Winning access is sampled in the reordering queue
- Reordering mechanism is handled between valid requests that reside in the reordering queue to select the access that will be dispatched to the DDR.
  - The reordering is held in order to optimize the accesses and to maximize the utilization of the DDR bus
  - As soon as the reordered access is completed (indicated by end of response or data phase) then it is erased from the associated queue and the MMDC is ready to receive the next available access from the master

In general, the reordering/arbitration mechanism is based on dynamic priority mechanism, which compares dynamic priorities between valid entries in the reordering queue and issues the entry with highest dynamic priority towards the DDR Logic.

The selection of the winning access is based on two modes, which can be activated together, as following:

- Real time channel mode:
  - Accesses with QoS='f' (i.e. awqos[3:0]/arqos[3:0] = "f") will bypass all other requests towards the DDR
- Dynamic scoring mode:
  - The arbitration mechanism is based on dynamic priority. Relevant for the accesses with QoS smaller than 'f' or when real time channel mode is disabled.

### **NOTE**

Due to re-ordering and optimization mechanism (per different AXI ID), the transactions towards the DDR may be driven in a different ID order they were received by the AXI master. In similar way, the write response, read response or read data may be driven to the AXI master in a different ID order.

#### **32.5.1.2 Real time channel mode**

When real time mode is enabled (i.e MAARCR[ARCR\_RCH\_EN] = "1") , all requests with QoS='f' (i.e. awqos[3:0]/aqos[3:0] = "f") will bypass all other pending accesses towards the DDR. This mode is enabled by default.

#### **32.5.1.3 Dynamic scoring mode (Arbitration Winning Conditions)**

The arbitration between pending accesses in the MMDC is handled according to a dynamic priority of each access.

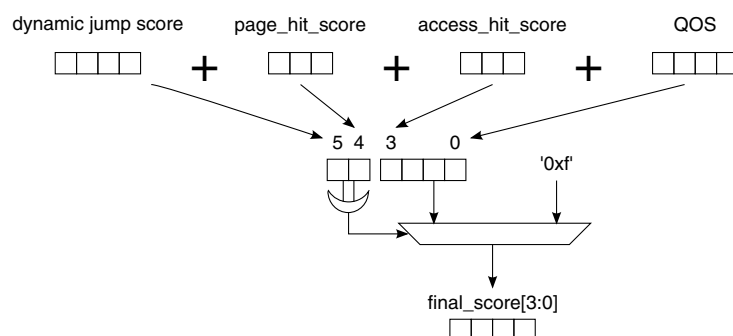
The dynamic priority (may be also called score) is calculated according to a sum of some factors (final\_score[3:0]), where part of them may be updated dynamically. The following will specify each scoring factor:

- MAARCR[ARCR\_PAG\_HIT] (Page hit score) - A static score which is taken into account in case the pending access has a page hit
- MAARCR[ARCR\_ACC\_HIT] (Access hit score) - A static score, which is taken into account in case the current access type (read/write) is the same as the access that has been dispatched to the DDR previously

- MAARCR[ARCR\_DYN\_JMP] (Dynamic jump score) - A dynamic score which is given to any pending access in case it was not chosen in the arbitration. The dynamic jump counter is limited by maximum value which is set in MAARCR[ARCR\_DYN\_MAX] .
- QoS score which is indicated through a sideband 4bits AXI signals (awqos[3:0]/ aqos[3:0]) and is driven by the AXI master per access

Note: In order to prevent an overflow in the total sum of scores, a clipping is held and selects the maximum score value of 'f' once a total scores sum is greater than 'f'.

The figure below shows the dynamic score calculations



**Figure 32-6. Dynamic score/priority calculation**

### 32.5.1.4 Guarding (aging) mechanism

The guarding mechanism (may be also called aging) is used to prevent a starvation of accesses.

As soon as the dynamic jump score reaches its maximum value (MAARCR[ARCR\_DYN\_MAX] ) then each time a pending request was not chosen in the arbitration, the "guarding" counter is incremented by 1. When the "guarding" counter reaches its predefined value, set in MAARCR[ARCR\_GUARD], the associated request gets the highest priority and will be chosen in the next arbitration cycle towards the DDR unless a real time channel (i.e access with QoS ="f") is arrived.

Note: In case real time channel has arrived then the dynamic score of the non real time channels won't increment in order to prevent a case where the "guarding" counter of more than one access has reached its limit.

## 32.5.2 Prediction mechanism

When prediction mechanism is enabled (i.e by configuring MDMISC[MIF3\_MODE]) then the MMDC predicts the chip-select, bank address and row address that is going to be issued towards the DDR before the access is physically dispatched towards DDR device.

That mechanism enables to prepare the DDR device with future accesses and improves the overall DDR performance.

This prediction mechanism operates in parallel to the reordering mechanism and may yield a prediction based on 3 levels of pending accesses:

1. Access in first stage of pipeline.
2. Valid access on AXI bus either read channel or write channel.
3. Valid access on special bus from arbitration - this access is chosen by the arbitration as the next miss access in its buffers

## 32.5.3 Special Optimization for accesses towards DDR3

In case an AXI read/write wrap non-aligned access is acknowledged in DDR3 mode with the same wrap boundary as the DDR wrap boundary then the MMDC will make an optimization and issue only one access towards the DDR, although all the accesses towards the DDR3 must be aligned.

For example: AXI write access with size of 128bits (awsize[2:0]=3'b100), length of 4 (awlen[3:0]=4'b0011) towards DDR3 x64 (burst length 8). In that case the AXI wrap boundary is 16Bx4=64B (0x40) and the DDR3 wrap boundary is 8Bx8=64B (0x40). If, for example, the AXI access is towards AXI address with suffix of 0x10 (non-aligned to 64B boundary) then the MMDC will get from the AXI master the data that is associated with addresses 0x10, 0x20, 0x30, 0x0. The MMDC will rearrange internally the data so it will match DDR3 alignment as following: 0x0, 0x10, 0x20, 0x30 and drive it in one access towards the DDR to address 0x0. The alternative was to issue two accesses towards the DDR with address 0x0 with different data masking

### NOTE

In read wrap access the same optimization is handled, while as soon as the critical AXI word is fetched from the DDR then it is driven immediately to the AXI master without buffering. Based on the example above, the master expects to fetch first the data that is associated with address 0x10. Therefore the MMDC will issue read access from address 0x0 of the DDR and as soon as the data that is associated with address 0x10 is received then it



will be driven back immediately to the master even before fetching the data of the further addresses.

## 32.6 MMDC Debug

### 32.6.1 Hardware debug monitor

The MMDC has a hardware debugging mechanism that monitors each access that is driven to the MMDC.

Every time this mechanism is enabled (setting of MADPCR0[DBG\_EN] to "1") then each access that will be dispatched to the DDR will be also observed in the I/O pads (i.e. over ipp\_do\_ddr\_debug[50:0]). The content of this bus is described in the table below.

**Table 32-9. Hardware monitor debugging**

Signal Name	Number of Bits	Description
acc_addr	[31:0]	AXI ADDRESS of the selected access
acc_type	1	access type of the selected access. "0" indicates write. "1" indicates read.
acc_id	[15:0]	AXI transaction ID of the selected access
valid_strobe	1	indication for a valid request . This signal will be asserted for 1 clock cycle

The fields above are organized as following:

MMDC\_DEBUG[50:0] = { 1'b0,valid\_strobe,acc\_id,access\_type,addr }

These signals are sent to IOMUX, in IOMUX user can configure it to be output from the chip for debug usage.

### 32.6.2 Step By Step (SBS) software monitor

The MMDC has a Step By Step (SBS) software debugging mechanism that monitors each access that is driven to the MMDC.

Every time this mechanism is triggered then one AXI access will be dispatched to the DDR and in parallel its attributes will be observed in a status register.

Once the "step by step" is enabled (i.e. MADPCR0[SBS\_EN] is "1") then all accesses to the DDR device will be halted.

Setting MADPCR0[SBS] to "1" will dispatch the access that is pending in the head of the MMDC queue (read or write). Upon every setting of MADPCR0[SBS]:

- The AXI attributes of the access will be sampled in the associated MASBS0 and MASBS1 fields
- MADPCR0[SBS] will be cleared automatically.

Setting again MADPCR0[SBS] to "1" will dispatch the next pending access in the MMDC queue.

## 32.7 MMDC Profiling

The profiling mechanism provides the ability to calculate the DDR utilization together with read and write accesses statistics towards DDR per given period of time.

MMDC supports the following profiling counters:

- MADPSR0 (Total cycles count) - Indicates the total amount of cycles of the profiling period (up to  $2^{32}$  cycles)
- MADPSR1 (Busy cycles count) - Indicates the total busy cycles during the profiling period
- MADPSR2 (Total read accesses count) - Indicates the total read accesses towards MMDC during the profiling period
- MADPSR3 (Total write accesses count) - Indicates the total write accesses towards MMDC during the profiling period
- MADPSR4 (Total read bytes count) - Indicates total bytes that were read from MMDC during the profiling period
- MADPSR5 (Total write bytes count) - Indicates total bytes that were written to MMDC during the profiling period

All profiling items described above are disabled by default. The following describes how to control the profiling mechanism:

- MADPCR0[DBG\_EN] enables profiling.
- MADPCR0[PRF\_FRZ] stops/freezes the profiling for example in case user wishes to perform DDR profiling per specific task. In order to resume profiling then MADPCR0[PRF\_FRZ] should be cleared.
- MADPCR0[DBG\_RST] clears all profiling counters
- MADPCR0[CYC\_OVF] indicates whether an overflow occurred in the total cycles counter (i.e. total amount of cycles are greater than  $2^{32}$ ). This field can only be cleared by writing '0'.

Read/Write statistics can be collected per specific AXI ID (16bits). The following fields in MADPCR1 register determines which AXI-ID or AXI-ID's to monitor:

- PRF\_AXI\_ID defines which AXI IDs are taken for profiling. Default value is 16'h0.
- PRF\_AXI\_ID\_MASK defines which bits from PRF\_AXI\_ID will be compared with AXI ID of read/write access. "1" means to monitor the associated bit and "0" means don't care. Default value is 16'h0000, meaning all IDs are monitored

So the AXI-IDs to be monitored are calculated according to the following equation:

$$(AXI-ID \& PRF\_AXI\_ID\_MASK) \text{ Xnor } (PRF\_AXI\_ID \& PRF\_AXI\_ID\_MASK)$$

For example if AXI ID's between A100 till A1FF are wished to be monitored then the following should be configured:

- PRF\_AXI\_ID = A100
- PRF\_AXI\_ID\_MASK = FF00

## 32.8 LPDDR2 Refresh Rate Update and Timing Derating

LPDDR2 devices may have a temperature sensor that is used to determine an appropriate refresh rate and whether AC timing derating is required. The status of the temperature sensor can be read through MRR command from LPDDR2 MR4 register.

The MMDC supports refresh update and timing derating mechanism on the fly. The following specify how to use that mechanism:

- Perform periodic polling on MR4 LPDDR2 register using MRR command
- Read MDMRR register and analyze the MR4 indication
- In case refresh rate update and/or AC timing derating is required then it is needed to update MDREF and/or MDMR4[tRCD\_DE, tRC\_DE, tRAS\_DE, tRP\_DE, tRRD\_DE] parameters

### NOTE

MDMR4[tRCD\_DE, tRC\_DE, tRAS\_DE, tRP\_DE, tRRD\_DE]  
are referred to the associated values configured at  
MDCFG3LP[tRC\_LP, tRP\_LP, tRCD\_LP], MDCFG1[tRAS],  
MDCFG2[tRRD]

- Assert MDMR4[UPDATE\_DE\_REQ]
- When the MMDC switch to the new values then an acknowledge will be indicated at MDMR4[UPDATE\_DE\_ACK]

## 32.9 DLL Off mode

DLL Off mode is supported only in DDR3 and let operating the DDR in low frequency (i.e. below 125MHz as defined in JEDEC standard).

For further details refer to DLL-off Mode chapter in the standard.

The following steps should be executed in order to switch from DLL on to DLL off mode:

- Assert CON\_REQ signal and wait to CON\_ACK assertion.
- Disable power down timers that can conflict with this sequence, such as: MAPSR[PSD], MDPDC[PWDT\_1], MDPDC[PWDT\_0], MDPDC[PRCT\_0], MDPDC[PRCT\_1].
- Execute precharge all banks command (via MDSCR).
- Execute MRW command to MR1 and disable RTT Nom (A9,A6,A2 =0) and DLL ON (A0 =1).
- Execute MRW command to MR2 in order to update CWL to 6.
- Execute MRW command to MR0 in order to update CL to 6.
- De-assert CON\_REQ signal.
- Enter self refresh mode. For further information refer to [Self refresh and Frequency change entry/exit](#).
- At self refresh entry acknowledge , Change to the desired frequency.
- Exit self refresh mode.
- Assert CON\_REQ and wait to CON\_ACK assertion.
- Enable Pull Down resistors on DQS (through the I/O-MUX ).
- Configure the MMDR register as following:
- Update tCWL =6 and tCL =6 to meet the values configured in the DDR device. (MDCFG0, MDCFG1)
- Disable ODT resistor (i.e. set MPODTCTRL to "0").
- Disable DQS gating (i.e. set MPDGCTRL0[DG\_DIS] to "1").
- Enable required power down timers that were disabled, such as: MAPSR[PSD], MDPDC[PWDT\_1], MDPDC[PWDT\_0], MDPDC[PRCT\_0], MDPDC[PRCT\_1].
- De-assert CON\_REQ and wait for de-assertion of CON\_ACK.

The following steps should be executed in order to switch from DLL off to DLL on:

- Execute precharge all banks command (via MDSCR).
- Enter self refresh mode. For further information refer to [Self refresh and Frequency change entry/exit](#).
- At self refresh entry acknowledge , Change to the desired frequency.
- Exit self refresh mode

- Assert CON\_REQ and wait to CON\_ACK assertion.
- Disable power down timers that can conflict with this sequence, such as: MAPSR[PSD], MDPDC[PWDT\_1], MDPDC[PWDT\_0], MDPDC[PRCT\_0], MDPDC[PRCT\_1].
- Execute MRW command to MR1 and enable RTT Nom (A9,A6,A2 ) and DLL ON (A0 =0 ).
- Execute MRW command to MR0 to reset the DLL (A8) and update CL value
- Execute MRW command to MR2 in order to update CWL value.
- Execute ZQ commnad.
- Reconfigure MMDC BLOCK
- Update tCWL and tCL to meet the values configured to the memory. (MDCGFG0, MDCFG1)
- Enable ODT resistor (i.e. MPODTCTRL register)
- Enable DQS gating (i.e. set MPDGCTRL0[DG\_DIS] to "0").
- Disable Pull Down resistors on DQS (through the I/O-MUX ).
- Enable required power down timers that were disabled, such as: MAPSR[PSD], MDPDC[PWDT\_1], MDPDC[PWDT\_0], MDPDC[PRCT\_0], MDPDC[PRCT\_1].
- De-assert CON\_REQ and wait for de-assertion of CON\_ACK.

## 32.10 ODT Configuration

The MMDC supports two DRAM\_ODT signals (DRAM\_ODT for each DRAM\_CS) in DDR3 mode in order to allow the DDR device to turn on/off its termination resistors. The MMDC suggests various configuration for the assertion of the ODT signals as well as configuration of several related timing.

The following specifies the options for configuring the assertion of DRAM\_ODT signals:

- Assert DRAM\_ODT signal for the non active DRAM\_CS in write. For example : when this bit asserted - if writing to DRAM\_CS0 the DRAM\_ODT of DRAM\_CS1 will be asserted. This is done by setting MPODTCTRL[0] to "1"
- Assert DRAM\_ODT signal for the active DRAM\_CS in write by by setting MPODTCTRL[1] to "1"
- Assert DRAM\_ODT signal for the non active DRAM\_CS in read by by setting MPODTCTRL[2] to "1"
- Assert DRAM\_ODT signal for the active DRAM\_CS in read by by setting MPODTCTRL[3] to "1"

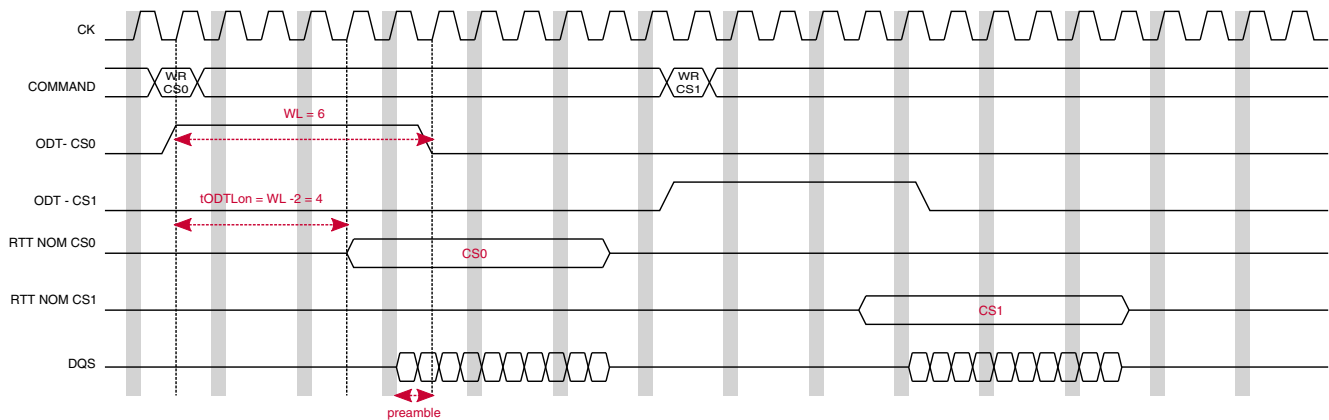
MDOTC register controls the timing for the DRAM\_ODT signals assertion.

**NOTE**

tODTLon determines the delay between DRAM\_ODT signal and the associated RTT, where according to JEDEC standard it equals WL(write latency) - 2. Therefore, the value configured to MDOTC[tODTLon] field should correspond with the value configured to MDCGFG1[tCWL].

In precharge power down mode, when all banks are closed, the assertion of ODT corresponds with tAOFPD and tAONPD which are configured in MDOTC register.

The figure below shows timing diagram of DRAM\_ODT and RTT signals while MPODTCTRL[0] is set to "1" (i.e. assertion of DRAM\_ODT to the non active DRAM\_CS in write access command) and MDOTC[tODTLon] is set to 4.



**Figure 32-7. ODT - Timing Diagram DDR3 WL=6, BL=8**

## 32.11 Calibration Process

The MMDC offers various calibration processes that are used to obtain better timing accuracy, board skew compensation and I/O pad driving strength adjustment.

Each calibration process can be performed either automatically (hardware) or manually (software), though the manual method is typically reserved for debugging purposes. The following calibration processes are supported:

**NOTE**

Power saving features should be disabled before the calibration process begin. (Such as: MDPDC[PWDT#], MDPDC[PRCT#], MAPSR[PSD])

- ZQ calibration for external DDR device (in DDR3 through ZQ calibration command and in LPDDR2 through MRW command)
  - Can be handled automatically for ZQ Short (periodically) and ZQ Long (at exit from self-refresh)
  - Can be handled manually at ZQ INIT
- ZQ calibration for i.MX DDR I/O pads for calibrating the DDR driving strength
  - The sequence can be handled automatically by hardware
  - The sequence can be handled step by step manually by software
- Read DQS gating calibration for DDR3 only. Adjustment of DQS gate with read preamble window. For further information refer to [Read DQS Gating Calibration](#)
- Read data calibration. Adjustment of read DQS with read data byte. For further information refer to [Read Calibration](#)
- Write data calibration. Adjustment of write DQS with write data byte. For further information refer to [Write Calibration](#)
- Write leveling calibration. Adjustment of write DQS with CK (DDR differential clock). For further information refer to [Write leveling Calibration](#)
- Read fine tuning. Adjustment of up to 7 delay-line units for each read data bit.
- Write fine tuning. Adjustment of up to 3 delay-line units for each read data bit.

### NOTE

Before starting any calibration process that involves the DDR3 device MPR mode or write leveling calibration, the following should be done:

- Disable the periodic refresh scheme (i.e. setting MDREF[REF\_SEL] = "00") and then issue manual refresh command burst by configuring MDSCR[CMD]= 0x2. At the end of the calibration it is needed to enable the periodic refresh scheme.
- Disable the automatic power saving mode (i.e set MAPSR[PSD] = "1").

## 32.11.1 delay-line

Each of the calibration processes controls several delay-lines for aligning data and strobes.

By default the delay-line is configured to generate 1/4 clock cycle of delay. The maximum delay that may be issued by the delay-line, while configured to the value 127, is as following:

- In BCS, -40C, 1.21V - 1.6ns.
- In WCS, 125C, 0.99V - 3.8ns

Moreover, when the operating clock is at the maximum allowed frequency, as appeared in the features list, then the delay-line is capable to issue a configurable delay of up to 1/2 clock cycle.

### **NOTE**

At the beginning of the calibration process the initial value of the delay-line must be a valid value (i.e. the strobes must be somewhere among the associated data window) though it might not be the optimal value. The delay-line calibration should be done after Read DQS gating and write-leveling calibrations.

In order to generate an adequate delay during normal operation of the MMDC the delay-line is going through an automatic measurement process during the refresh period of the DDR device

## **32.11.2 ZQ calibration**

The MMDC supports ZQ calibration process to calibrate the driving strength of the i.MX DDR I/O pads as well as driving ZQ commands to calibrate the external DDR device driving strength.

The first i.MX ZQ calibration (after booting the processor) is performed prior to turning on the MMDC. Subsequent i.MX ZQ calibrations may be executed in parallel to the DDR ZQ calibration. The MMDC supports 2 types of ZQ calibration commands: short and long.

The ZQ long calibration is executed during power up sequence, when existing self-refresh mode or when exiting slow precharge power down (DLL lock can be done in parallel). The ZQ short calibration is executed periodically according to a configurable timer defined by MPZQHWCTRL[ZQ\_HW\_PER].

The field MPZQHWCTRL[ZQ\_MODE] determines whether the MMDC will execute ZQ calibration to i.MX DDR I/O pads and/or issue ZQ short/long command to the DDR device.

The MMDC supports both automatic (hardware) and manual (software) ZQ calibration process for the i.MX DDR I/O pads.

It is possible to perform automatic (hardware) ZQ calibration only once (i.e. non-periodical) by asserting MPZQHWCTRL[ZQ\_HW\_FOR].



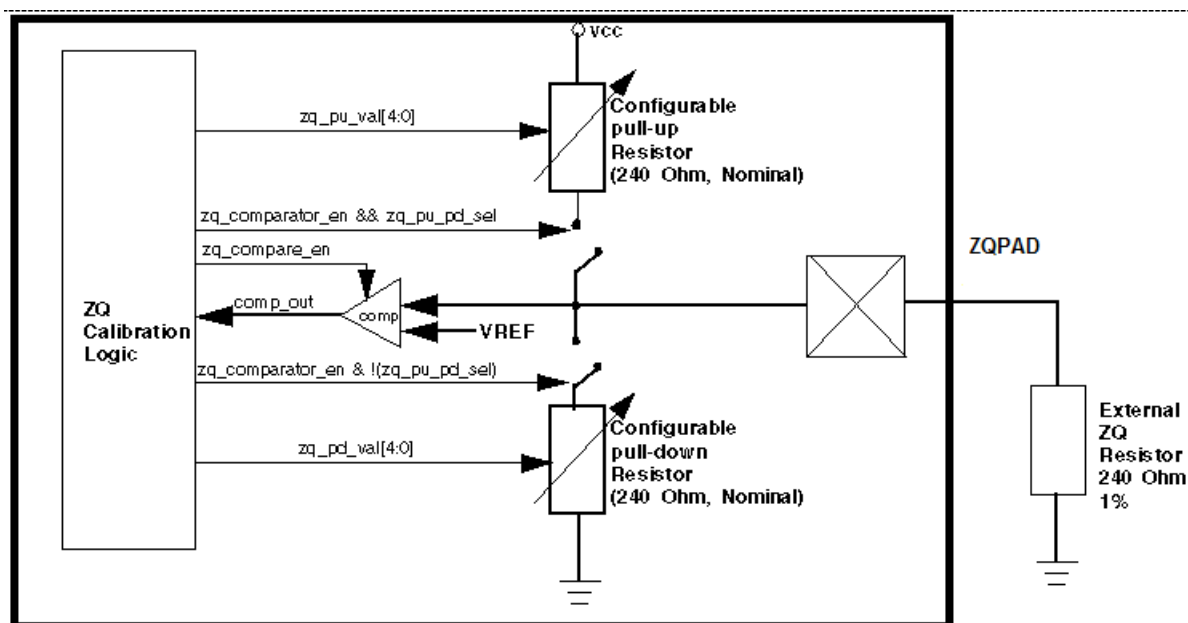


Figure 32-8. MMDC ZQ IF with PAD

### 32.11.2.1 ZQ automatic (hardware) calibration process

The ZQ automatic calibration lasts 11 steps. 5 steps for the pull up resistors calibration and 6 for the pull down resistors calibration.

The calibration control interface with ZQ pin is described below:

The calibration process is as follows:

#### 32.11.2.1.1 ZQ automatic Pull-up calibration

The MMDC perform a handshaking mechanism with the i.MX ZQ calibration pad automatically

as following:

1. The MMDC drives `zq_comparator_en` to "1"
2. The MMDC waits few cycles according to `MPZQHWCTRL[EARLY_COMPARATOR_EN_TIMER]`
3. The MMDC drives `zq_pu_pd_sel` to "1" for indication of pull-up calibration and drives `zq_pu_val[4:0] = 5'b00000`
4. MMDC drives `zq_pu_val[4]` to "1"
5. MMDC asserts `zq_compare_en`
6. MMDC waits few cycles according to `MPZQSWCTRL[ZQ_CMP_OUT_SMP]` before sampling the comparator output (i.e `zq_comp_out`). If `zq_comp_out` is "1"

then it means that the output voltage is greater than  $V_{dd}/2$  (i.e. internal resistor is less than 240 ohm) and drives bit `zq_pu_val[4]` to "1" else it drives `zq_pu_val[4]` to "0"

7. MMDC deasserts `zq_compare_en`
8. MMDC repeats steps 4- 7 for `zq_pu_val` bits 3 to 0
9. MMDC drives ZQ calibration result to `MPZQHWCTRL[ZQ_HW_PU_RES]`
10. MMDC advances to pull-down calibration

### 32.11.2.1.2 ZQ automatic Pull-down calibration

1. The MMDC drives `zq_pu_pd_sel` to "0" for indication of pull-down calibration and drives `zq_pd_val[4:0] = 5'b00000`
2. MMDC drives `zq_pd_val[4]` to "1"
3. MMDC asserts `zq_compare_en`
4. MMDC waits few cycles according to `MPZQSWCTRL[ZQ_CMP_OUT_SMP]` before sampling the comparator output (i.e `zq_comp_out`). If `zq_comp_out` is "1" then it means that the output voltage is greater than  $V_{dd}/2$  (i.e. internal resistor is less than 240 ohm) and drives bit `zq_pd_val[4]` to "0" else it drives `zq_pd_val[4]` to "1"
5. MMDC deasserts `zq_compare_en`
6. MMDC repeats steps 12- 15 for `zq_pd_val` bits 3 to 0
7. MMDC drives ZQ calibration result to `MPZQHWCTRL[ZQ_HW_PD_RES]`
8. MMDC deassert `zq_comparator_en` to indicate the completion of the ZQ calibration

### 32.11.2.2 ZQ software calibration process

The ZQ calibration can be done also in software. However since software ZQ calibration is much slower than hardware calibration it should be used mainly for debugging.

Software should configure the ZQ calibration parameters (Pull-up or Pull-down and their value) then assert the `MPZQSWCTRL[ZQ_SW_FOR]` bit. Then software should wait till `ZQ_SW_FOR` is de-asserted and use `ZQ_SW_RES` status bit in order to calculate the next ZQ calibration parameters.

### 32.11.2.3 ZQ calibration commands

Before the MMDC can issue a `ZQCL/ZQCS` command to the memory it should precharge all memory banks and wait `tRP` period. A single ZQ command can be issued to all devices as long as the devices don't share the same ZQ resistor.

When the MMDC issues the ZQ command it should also drive `A10` (long or short command) and `CS` (0, 1 or both).

The MMDC must keep the memory lines quiet (except for CK) for the ZQ calibration time as defined in the Jedec (512 cycles for ZQCL after reset, 256 for other ZQCL and 64 for ZQCS).

### 32.11.3 Read DQS Gating Calibration

The read DQS gating calibration is used to adjust the read DQS gating with the middle of the read DQS preamble.

The DQS gating includes a delay of up to 7 cycles (The delay is chosen according to two fields MPDGCTRL#[DG\_HC\_DEL#] and MPDGCTRL#[DG\_DL\_ABS\_OFFSET#]

Each DQS has its own delay-line. The DQS gating process can be done for all DQS in parallel.

#### NOTE

In LPDDR2 mode hardware Read DQS gating should be disabled and Pull-up/pull-down resistors on DQS/DQS# should be enabled while ODT resistors must be disconnected.

In DDR3\_x64 mode activation of the calibration is done by setting MPDGCTRL0[HW\_DG\_EN] at address 0xBASE0\_083C

#### 32.11.3.1 Hardware DQS Gating Calibration

- There are two modes of operations:
  - Calibration with the MPR (Multi Purpose Register)
  - Calibration with MMDC pre-defined values

##### 32.11.3.1.1 Hardware DQS Calibration with MPR

The following steps should be executed:

1. Precharge all active banks (Can be done through MDSCR) as required by the standard.
2. Enter the DDR device into MPR mode through MRS commands
3. Configure the MMDC to work with MPR mode by asserting MPPDCMPR2[MPR\_CMP]
4. Make sure that the initial value that is configured in the read delay line absolute offset of each byte (i.e. MPRDDLCTL[RD\_DL\_ABS\_OFFSET#]) will place the read DQS somewhere inside the read DQ window

5. Start the calibration process by asserting MPDGCTRL0[HW\_DG\_EN]

### 32.11.3.1.2 Hardware DQS Calibration with pre-defined value

In case pre-defined mode is used, (i.e. MPPDCMPR2[MPR\_CMP]) is cleared, then the following steps should be executed:

1. Precharge all active banks (Can be done through MDSCR) as required by the standard.
2. Configure the pre-defined value, which reflects the value that will be written and compared through the read calibration, to MPPDCMPR1[PDV1, PDV2]
3. Issue write access to the external DDR device by setting MPSWDAR0[SW\_DUMMY\_WR] = 1 (MMDC will generate internally write access without intervention of the system towards bank 0, row 0, column 0)
4. Make sure that the initial value that is configured in the read delay line absolute offset of each byte (i.e. MPRDDLCTL[RD\_DL\_ABS\_OFFSET#] ) will place the read DQS somewhere inside the read DQ window
5. Start the calibration process by asserting MPDGCTRL0[HW\_DG\_EN]

The following steps will be executed automatically by the MMDC for both modes (MPR and Pre-defined value):

6. MMDC waits till the read DQS delay-line is updated with the absolute delay value for all bytes at MPDGCTRL#[DG\_HC\_DEL#] and MPDGCTRL#[DG\_DL\_ABS\_OFFSET#] and also satisfying the Tmod + 4 requirement
7. MMDC drives read command to the external DDR devices and waits 16 or 32 cycles (according to MPDGCTRL0[DG\_CMP\_CYC] assuming that the data has arrived from the DDR device.
8. MMDC compares the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the read DQS gating is asserted in illegal time point. If the comparison passes then MMDC advances to step 14
9. MMDC resets the read FIFO (to the inverted pre-defined/MPR value) and it's pointers by setting MPDGCTRL[RST\_RD\_FIFO] = 1
10. MMDC increments the read DQS gating delay of each byte by half cycle (i.e. MPDGCTRL#[DG\_HC\_DEL#] + 1)
11. MMDC drives read command to the external DDR devices and waits 16 or 32 cycles (according to MPDGCTRL0[DG\_CMP\_CYC] assuming that the data has arrived from the DDR device.
12. MMDC compares the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8).

13. If the comparison fails then it indicates that the read DQS gating is asserted in illegal time point and it is needed to repeat steps 9-12. If the comparison passes then MMDC stores the value of the temporary low boundary and advances to next step
14. MMDC increments the read DQS gating delay-line of each byte by half cycle (i.e.  $\text{MPDGCTRL}\#[\text{DG\_HC\_DEL}\#] + 1$ ) and issue measurement process of the read DQS gating delay-line to update itself with the new value
15. MMDC drives read command to the external DDR devices and waits 16 or 32 cycles (according to  $\text{MPDGCTRL0}[\text{DG\_CMP\_CYC}]$  assuming that the data has arrived from the DDR device.
16. MMDC compares the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8)
17. If the comparison passes then it indicates that the read DQS gating is asserted inside the read preamble window and it is needed to repeat steps 14-16. If the comparison fails then MMDC stores the value of the temporary upper boundary and starts searching the adequate low and high boundaries
18. MMDC returns to the temporary low boundary minus half cycle and issue measurement process of the read DQS gating delay-line to update itself with the new value
19. MMDC drives read command to the external DDR devices and waits 16 or 32 cycles (according to  $\text{MPDGCTRL0}[\text{DG\_CMP\_CYC}]$  assuming that the data has arrived from the DDR device.
20. MMDC compares the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8)
21. If the comparison fails then it indicates that the read DQS gating is asserted in illegal time point and it is needed to repeat steps 22-23. If the comparison passes then MMDC stores the value of the adequate low boundary and advances to step 24
22. MMDC resets the read FIFO (to the inverted pre-defined/MPR value) and it's pointers by setting  $\text{MPDGCTRL}[\text{RST\_RD\_FIFO}] = 1$
23. MMDC increments the read DQS gating delay of each byte by 1 (i.e.  $\text{MPDGCTRL}\#[\text{DG\_DL\_ABS\_OFFSET}\#] + 1$ ) and issue measurement process of the read DQS gating delay-line to update itself with the new value and advances to step 19
24. MMDC returns to the temporary upper boundary minus half cycle and issue measurement process of the read DQS gating delay-line to update itself with the new value
25. MMDC drives read command to the external DDR devices and waits 16 or 32 cycles (according to  $\text{MPDGCTRL0}[\text{DG\_CMP\_CYC}]$  assuming that the data has arrived from the DDR device.
26. MMDC compares the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8)

27. If the comparison passes then it indicates that the read DQS gating is asserted inside the read preamble window and it is needed to repeat steps 28-29. If the comparison fails then MMDC stores the value minus 1 of the adequate upper boundary and advances to step 30
28. MMDC resets the read FIFO (to the inverted pre-defined/MPR value) and it's pointers by setting `MPDGCTRL[RST_RD_FIFO] = 1`
29. MMDC increments the read DQS gating delay of each byte by 1 (i.e. `MPDGCTRL#[DG_DL_ABS_OFFSET#] + 1`) and issue measurement process of the read DQS gating delay-line to update itself with the new value and advances to step 25
30. After the MMDC finds the window boundary (lower and upper) of each read data byte then it stores the average between lower and upper boundaries at the associated `MPDGCTRL#[DG_DL_ABS_OFFSET#]` and issue measurement process of the read DQS delay-line to update itself with the new value.
31. MMDC indicates that the read DQS gating calibration had finished by setting `MPDGCTRL0[HW_DG_EN] = 0`
32. Exit the DDR device from MPR mode through MRS command
33. Read the upper boundary that was found: `MPDGHWST#[HW_DG_UP#]`. This field is 11 bits, 7 LSB bits correspond to `MPDGCTRL#[DG_DL_ABS_OFFSET#]` upper limit value and 4 MSB bits correspond to `MPDGCTRL#[DG_HC_DEL#]` upper limit value.
34. Set `MPDGHWST#[HW_DG_UP#[6:0]]` to `MPDGCTRL#[DG_DL_ABS_OFFSET#]`.
35. Set `(MPDGHWST#[HW_DG_UP#[10:7]] - 1)` to `MPDGCTRL#[DG_HC_DEL#]`. (We set the DQS gating value to be the upper limit value minus 1 half cycle)

### 32.11.3.2 SW read DQS gating Calibration

- There are two modes of operations:
- Calibration with the MPR (Multi Purpose Register)
- Calibration with MMDC pre-defined values

#### 32.11.3.2.1 SW read Calibration with MPR

The following steps should be executed:

1. Precharge all active banks (Can be done through MDSCR) as required by the standard.
2. Enter the DDR device into MPR mode through MRS commands
3. Configure the MMDC to work with MPR mode by asserting `MPPDCMPR2[MPR_CMP]`

4. Make sure that the initial value that is configured in the read delay line absolute offset of each byte (i.e. `MPRDDLCTL[RD_DL_ABS_OFFSET#]` ) will place the read DQS somewhere inside the read DQ window

### 32.11.3.2.2 SW read Calibration with pre-defined value

In case pre-defined mode is used, (i.e. `MPPDCMPR2[MPR_CMP]`) is cleared, then the following steps should be executed:

1. Precharge all active banks (Can be done through `MDSCR`) as required by the standard.
2. Configure the pre-defined value, which reflects the value that will be written and compared through the read calibration, to `MPPDCMPR1[PDV1, PDV2]`
3. Issue write access (with any legal DDR address) to external DDR device
4. Make sure that the initial value that is configured in the read delay line absolute offset of each byte (i.e. `MPRDDLCTL[RD_DL_ABS_OFFSET#]` ) will place the read DQS somewhere inside the read DQ window

The following steps should be executed automatically by the MMDC for both modes (MPR and Pre-defined value):

5. Configure the read DQS delay-line to issue zero delay by setting `MPDGCTRL#[DG_DL_ABS_OFFSET#] = 0` and `MPDGCTRL#[DG_HC_DEL#] = 0`
6. Force the delay line to measure itself and to issue the requested read delay by configuring `MPMUR[FRC_MSR] = 1`
7. Wait 16 DDR cycles till the read DQS delay-line is updated with the absolute delay value for all bytes
8. Issue read command (with the legal DDR address chosen in step 3) from the external DDR device
9. Waits 16 or 32 cycles (according to `MPDGCTRL0[DG_CMP_CYC]`) assuming that the data has arrived from the DDR device.
10. Compare the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the read DQS gating is asserted in illegal time point. If the comparison passes then advance to step 15
11. MMDC resets the read FIFO (to the inverted pre-defined/MPR value) and it's pointers by setting `MPDGCTRL[RST_RD_FIFO] = 1`
12. Increment the read DQS gating delay of each byte by half cycle (i.e. `MPDGCTRL#[DG_HC_DEL#] + 1`)
13. MMDC drives read command to the external DDR devices and waits 16 or 32 cycles (according to `MPDGCTRL0[DG_CMP_CYC]`) assuming that the data has arrived from the DDR device.

14. Compare the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the read DQS gating is asserted in illegal time point and it is needed to repeat steps 11 - 14. If the comparison passes then advance to step 15
15. Store the temporary lower boundary and start searching the temporary upper boundary
16. MMDC resets the read FIFO (to the inverted pre-defined/MPR value) and it's pointers by setting `MPDGCTRL[RST_RD_FIFO] = 1`
17. Increment the read DQS gating delay of each byte by half cycle (i.e. `MPDGCTRL#[DG_HC_DEL#] + 1`)
18. MMDC drives read command to the external DDR devices and waits 16 or 32 cycles (according to `MPDGCTRL0[DG_CMP_CYC]` assuming that the data has arrived from the DDR device.
19. Compare the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8).
20. If the comparison passes then it indicates that the read DQS gating is asserted inside the read preamble window and it is needed to repeat steps 16-19. If the comparison fails then it is needed to store the value of the temporary upper boundary and starts searching the adequate low and high boundaries
21. Load the temporary low boundary minus half cycle into the associated `MPDGCTRL#[DG_HC_DEL#]`
22. Reset the read FIFO (to the inverted pre-defined/MPR value) and it's pointers by setting `MPDGCTRL[RST_RD_FIFO] = 1`
23. Increment the read DQS gating delay of each byte by 1 (i.e. `MPDGCTRL#[DG_DL_ABS_OFFSET#] + 1`) and force the delay line to measure itself and to issue the requested read DQS delay by configuring `MPMUR[FRC_MSR] = 1`
24. Issue read command to the external DDR devices and waits 16 or 32 cycles (according to `MPDGCTRL0[DG_CMP_CYC]`) assuming that the data has arrived from the DDR device.
25. Compare the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8)
26. If the comparison fails then it indicates that the read DQS gating is asserted in illegal time point and it is needed to repeat steps 22-26. If the comparisons passes then advance to the next step.
27. Store the adequate lower boundary
28. Load the temporary upper boundary minus half cycle into the associated `MPDGCTRL#[DG_HC_DEL#]`
29. Reset the read FIFO (to the inverted pre-defined/MPR value) and it's pointers by setting `MPDGCTRL[RST_RD_FIFO] = 1`



30. Increment the read DQS gating delay of each byte by 1 (i.e.  $\text{MPDGCTRL}\#[\text{DG\_DL\_ABS\_OFFSET}\#] + 1$ ) and force the delay line to measure itself and to issue the requested read DQS delay by configuring  $\text{MPMUR}[\text{FRC\_MSR}] = 1$
31. Issue read command to the external DDR devices and waits 16 or 32 cycles (according to  $\text{MPDGCTRL0}[\text{DG\_CMP\_CYC}]$  assuming that the data has arrived from the DDR device.
32. Compare the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8)
33. If the comparison passes then it is needed to repeat steps 29-32. If the comparisons fails then advance to the next step.
34. Reset the read FIFO (to the inverted pre-defined/MPR value) and it's pointers by setting  $\text{MPDGCTRL}[\text{RST\_RD\_FIFO}] = 1$
35. Store the adequate upper boundary.
36. Keep the  $\text{MPDGCTRL}\#[\text{DG\_DL\_ABS\_OFFSET}\#]$  value of the upper limit.
37. Set  $\text{MPDGCTRL}\#[\text{DG\_HC\_DEL}\#] = (\text{MPDGCTRL}\#[\text{DG\_HC\_DEL}\#] - 1)$ . (We set the DQS gating value to be the upper limit value minus 1 half cycle)
38. Issue the requested read DQS delay by configuring  $\text{MPMUR}[\text{FRC\_MSR}] = 1$
39. Exit the DDR device from MPR mode through MRS command

### 32.11.4 Read Calibration

The read calibration is used to adjust the read DQS with read data byte.

It is assumed that the read DQS gating calibration process is completed prior to the read calibration.

#### NOTE

In DDR3 mode, the activation of the calibration is done by setting  $\text{MPRDDLHWCTL}[\text{HW\_RD\_DL\_EN}]$  at address  $0x\text{BASE0\_0860}$  In LP2\_x16, LP2\_x32 the activation of the calibration is done by setting  $\text{MPRDDLHWCTL}[\text{HW\_RD\_DL\_EN}]$  at address  $0x\text{BASE0\_0860}$ .

#### 32.11.4.1 Hardware (automatic) Read Calibration

- There are two modes of operations:
- Calibration with the MPR (Multi Purpose Register)/DQ calibration(LPDDR2)
- Calibration with MMDC pre-defined values

### 32.11.4.1.1 Hardware (automatic) Calibration with MPR (DDR3) /DQ Calibration (LPDDR2)

The following steps should be executed:

1. Precharge all active banks (can be done through MDSCR) as required by the standard.
2. Enter the DDR device into MPR/DQ calibration mode through MRS/MRW commands.
3. Configure the MMDC to work with MPR/DQ calibration mode by asserting MPPDCMPR2[MPR\_CMP].
4. Make sure that the initial value that is configured in the read delay line absolute offset of each byte (MPRDDLCTL[RD\_DL\_ABS\_OFFSET#]) will place the read DQS somewhere inside the read DQ window.
5. Start the calibration process by asserting MPRDDLHWCTL[HW\_RD\_DL\_EN].

### 32.11.4.1.2 Hardware (automatic) Calibration with pre-defined value

In case pre-defined mode is used, i.e. MPPDCMPR2[MPR\_CMP] is cleared, then the following steps should be executed:

1. Precharge all active banks (Can be done through MDSCR) as required by the standard.
2. Configure the pre-defined value, which reflects the value that will be written and compared through the read calibration, to MPPDCMPR1[PDV1, PDV2]
3. Issue write access to the external DDR device by setting MPSWDAR0[SW\_DUMMY\_WR] = 1 (MMDC will generate internally write access without intervention of the system towards bank 0, row 0, column 0)
4. Make sure that the initial value that is configured in the read delay line absolute offset of each byte (i.e. MPRDDLCTL[RD\_DL\_ABS\_OFFSET#] ) will place the read DQS somewhere inside the read DQ window
5. Start the calibration process by asserting MPRDDLHWCTL[HW\_RD\_DL\_EN]

The following steps will be executed automatically by the MMDC for both modes (MPR and Pre-defined value):

6. MMDC waits till the read delay-line is updated with the absolute delay value for all bytes at MPRDDLCTL[RD\_DL\_ABS\_OFFSET#] and also satisfying the Tmod + 4 requirement
7. MMDC drives read command to the external DDR devices and waits 16 or 32 cycles (according to MPRDDLHWCTL[HW\_RD\_DL\_CMP\_CYC]) assuming that the data has arrived from the DDR device.

8. MMDC compares the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the initial read DQS isn't inside the read DQ window and the MMDC generates an error for the associated byte at MPRDDLHWCTL[HW\_RD\_DL\_ERR#] . If the comparison passes then MMDC advances to next step.
9. MMDC resets the rd fifo (to the inverted pre-defined/MPR value) and it's pointers by setting MPDGCTRL[RST\_RD\_FIFO] = 1
10. MMDC decrements the read delay line absolute offset of each byte by 1 (i.e. MPRDDLCTL[RD\_DL\_ABS\_OFFSET#] ) and issue measurement process of the read delay-line to update itself with the new value.
11. MMDC drives read command to the DDR external devices and waits 16 or 32 cycles (according to MPRDDLHWCTL[HW\_RD\_DL\_CMP\_CYC]) assuming that the data has arrived from the DDR device
12. MMDC compares the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it stores the low read boundary of the associated byte for each byte at MPRDDLHWST0/1[HW\_RD\_DL\_LOW#] . If the comparison passes then MMDC repeats steps 9-11. If all read data comparisons fail then the MMDC advances to the next step
13. The MMDC start seeking the upper boundary and sets the read delay line absolute offset of each byte to the initial value + 1 as determined at step 4 and issue measurement process of the read delay-line to update itself with the new value
14. MMDC resets the rd fifo (to the inverted pre-defined value) and it's pointers by setting MPDGCTRL[RST\_RD\_FIFO] = 1
15. MMDC drives read command to the DDR external devices and waits 16 or 32 cycles (according to MPRDDLHWCTL[HW\_RD\_DL\_CMP\_CYC]) assuming that the data has arrived from the DDR device
16. MMDC compares the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it stores the upper read boundary of the associated byte for each byte at MPRDDLHWST0/1[HW\_RD\_DL\_UP#] . If the comparison passes then MMDC increments the read delay line absolute offset of each byte by 1 (i.e. MPRDDLCTL[RD\_DL\_ABS\_OFFSET#] ) and issue measurement process of the read delay-line to update itself with the new value.
17. If all read data comparisons fail then the MMDC advances to the next step. otherwise, MMDC repeats steps 14-16.
18. After the MMDC finds the window boundary (lower and upper) of each read data byte then it stores the average between lower and upper boundaries at the associated MPRDDLCTL[RD\_DL\_ABS\_OFFSET#] and issue measurement process of the read delay-line to update itself with the new value.

19. MMDC indicates that the read data calibration had finished by setting `MPRDDLHWCTL[HW_RD_DL_EN] = 0`
20. Exit the DDR device from MPR/DQ calibration mode through MRS/MRW commands

### 32.11.4.2 SW Read Calibration

- There are two modes of operations:
- Calibration with the MPR (Multi Purpose Register)/DQ calibration(LPDDR2)
- Calibration with MMDC pre-defined values

#### 32.11.4.2.1 Calibration with MPR(DDR3)/DQ calibration(LPDDR2)

The following steps should be executed:

1. Precharge all active banks (Can be done through MDSCR) as required by the standard.
2. Enter the DDR device into MPR/DQ calibration mode through MRS/MRW commands
3. Configure the MMDC to work with MPR/DQ calibration mode by asserting `MPPDCMPR2[MPR_CMP]`
4. Make sure that the initial value that is configured in the read delay line absolute offset of each byte (i.e. `MPRDDLCTL[RD_DL_ABS_OFFSET#]` ) will place the read DQS somewhere inside the read DQ window

#### 32.11.4.2.2 Calibration with pre-defined value

In case pre-defined mode is used, i.e. `MPPDCMPR2[MPR_CMP]` is cleared, then the following steps should be executed:

1. Precharge all active banks (Can be done through MDSCR) as required by the standard.
2. Configure the pre-defined value, which reflects the value that will be written and compared through the read calibration, to `MPPDCMPR1[PDV1, PDV2]`
3. Issue write access (with any legal DDR address) to external DDR device.
4. Make sure that the initial value that is configured in the read delay line absolute offset of each byte (i.e. `MPRDDLCTL[RD_DL_ABS_OFFSET#]` ) will place the read DQS somewhere inside the read DQ window

The following steps will be executed manually by SW for both modes (MPR/DQ calibration and Pre-defined value):

5. Force the delay line to measure itself and to issue the requested read delay by configuring `MPMUR[FRC_MSR] = 1`
6. Wait 16 DDR cycles till the read delay-line is updated with the absolute delay value for all bytes
7. Issue read command (with any legal DDR address) from the external DDR device
8. Compare the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the initial read DQS isn't inside the read DQ window. If the comparison passes then advance to next step.
9. Reset the rd fifo (to the inverted pre-defined/MPR value) and it's pointers by setting `MPDGCTRL[RST_RD_FIFO] = 1`
10. Decrement the read delay line absolute offset of each byte by 1 (i.e. `MPRDDLCTL[RD_DL_ABS_OFFSET#]` )
11. Force the delay line to measure itself and to issue the requested read delay by configuring `MPMUR[FRC_MSR] = 1`
12. Issue read command (with the legal DDR address chosen in step 7) from the external DDR device and waits 16 or 32 cycles (according to `MPRDDLHWCTL[HW_RD_DL_CMP_CYC]`) assuming that the data has arrived from the DDR device
13. Compare the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it is needed to store the low read boundary of the associated byte at of each byte . If the comparison passes then repeat steps 9-12. If all read data comparisons fail then advance to the next step.
14. Start seeking the upper boundary and set the read delay line absolute offset of each byte to the initial value + 1 as determined at step 4
15. Force the delay line to measure itself and to issue the requested read delay by configuring `MPMUR[FRC_MSR] = 1`
16. Resets the rd fifo (to the inverted pre-defined/MPR value) and it's pointers by setting `MPDGCTRL[RST_RD_FIFO] = 1`
17. Issue read command (with the legal DDR address chosen in step 7) from the external DDR device
18. Compare the read data byte to the associated byte in the pre-defined/MPR value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it is needed to store the upper read boundary of the associated byte at of each byte. If the comparison passes then increment the read delay line absolute offset of each byte by 1 (i.e. `MPRDDLCTL[RD_DL_ABS_OFFSET#]` )
19. Force the delay line to measure itself and to issue the requested read delay by configuring `MPMUR[FRC_MSR] = 1`
20. If all read data comparisons fail then advance to the next step, else repeat steps 16-19

21. After finding the window boundary (lower and upper) of each read data byte then calculate the average between lower and upper boundaries and store the associated average at MPRDDLCTL[RD\_DL\_ABS\_OFFSET#]
22. Force the delay line to measure itself and to issue the requested read delay by configuring MPMUR[FRC\_MSR] = 1
23. Exit the DDR device from MPR/DQ calibration mode through MRS/MRW commands.

## 32.11.5 Write Calibration

The write calibration is used to adjust the write DQS with write data byte. It is assumed that the read calibration process is completed prior to the write calibration.

### 32.11.5.1 HW (automatic) Write Calibration

The following steps should be executed:

1. Make sure that the initial value that is configured in the write delay line absolute offset of each byte (i.e. MPWRDLCTL[WR\_DL\_ABS\_OFFSET#] ) will place the write DQS somewhere inside the write DQ window
2. Configure the pre-defined value, which reflects the value that will be written and compared through the write calibration, to MPPDCMPR1[PDV1, PDV2]
3. Assert MPWRDLHWCTL0[HW\_WR\_DL\_EN]

The following steps will be executed automatically:

4. MMDC waits till the write delay-line is updated with the absolute delay value for all bytes at MPWRDCTL[WR\_DL\_ABS\_OFFSET#]
5. MMDC drives write command to the external DDR devices (to bank 0 address 0) and waits 16 or 32 cycles (according to MPWRDLHWCTL[HW\_WR\_DL\_CMP\_CYC]) assuming that the data has arrived to the DDR device.
6. MMDC drives read command to the same address from the external DDR
7. MMDC compares the read data byte to the associated byte in the pre-defined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the initial write DQS isn't inside the write DQ window and the MMDC generates an error for the associated byte at MPWRDLHWCTL[HW\_WR\_DL\_ERR#] . If the comparison passes then MMDC advances to next step.
8. MMDC resets the rd fifo (to the inverted pre-defined value) and it's pointers by setting MPDGCTRL[RST\_RD\_FIFO] = 1

9. MMDC decrements the write delay line absolute offset of each byte by 1 (i.e. `MPWRDLCTL[WR_DL_ABS_OFFSET#]` ) and issue measurement process of the write delay-line to update itself with the new value.
10. MMDC drives write command to the external DDR devices (to bank 0 address 0) and waits 16 or 32 cycles (according to `MPWRDLHWCTL[HW_WR_DL_CMP_CYC]`) assuming that the data has arrived to the DDR device
11. MMDC drives read command to the same address from the external DDR
12. MMDC compares the read data byte to the associated byte in the pre-defined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it stores the low write boundary of the associated byte of each byte at `MPWRDLHWST0/1[HW_WR_DL_LOW#]` . If the comparison passes then MMDC repeats steps 8-11. If all data comparisons fail then the MMDC advances to the next step
13. The MMDC start seeking the upper boundary and sets the write delay line absolute offset of each byte to the initial value + 1 as determined at step 4 and issue measurement process of the write delay-line to update itself with the new value
14. MMDC resets the rd fifo (to the inverted pre-defined value) and its pointers by setting `MPDGCTRL[RST_RD_FIFO] = 1`
15. MMDC drives write command to the external DDR devices (to bank 0 address 0) and waits 16 or 32 cycles (according to `MPWRDLHWCTL[HW_WR_DL_CMP_CYC]`) assuming that the data has arrived to the DDR device.
16. MMDC drives read command to the same address from the external DDR
17. MMDC compares the read data byte to the associated byte in the pre-defined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it stores the upper write boundary of the associated byte of each byte at `MPWRDLHWST0/1[HW_WR_DL_UP#]` . If the comparison passes then MMDC increments the write delay line absolute offset of each byte by 1 (i.e. `MPWRDLCTL[WR_DL_ABS_OFFSET#]` ) and issue measurement process of the write delay-line to update itself with the new value.
18. MMDC repeats steps 14-17. If all data comparisons fail then the MMDC advances to the next step
19. .After the MMDC finds the window boundary (lower and upper) of each write data byte then it stores the average between lower and upper boundaries at the associated `MPWRDLCTL[WR_DL_ABS_OFFSET#]` and issue measurement process of the write delay-line to update itself with the new value.
20. MMDC indicates that the write data calibration had finished by setting `MPWRDLHWCTL[HW_WR_DL_EN] = 0`

### 32.11.5.2 SW Write Calibration

The following steps should be executed:

#### NOTE

It is recommended to perform the write calibration using the HW method. The SW method is provided for debug purposes only.

1. Make sure that the initial value that is configured in the write delay line absolute offset of each byte (i.e. `MPWRDLCTL[WR_DL_ABS_OFFSET#]` ) will place the write DQS somewhere inside the write DQ window
2. Configure the pre-defined value, which reflects the value that will be written and compared through the write calibration, to `MPPDCMPR1[PDV1, PDV2]`
3. Force the delay line to measure itself and to issue the requested write delay by configuring `MPMUR[FRC_MSR] = 1`
4. Wait 16 DDR cycles till the write delay-line is updated with the absolute delay value for all bytes
5. Issue write command to any legal DDR address of the external DDR device
6. Issue read command, to the address written previously, from the external DDR device
7. Compare the read data byte to the associated byte in the pre-defined value for all bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it indicates that the initial write DQS isn't inside the write DQ window. If the comparison passes then advance to next step.
8. MMDC resets the rd fifo (to the inverted pre-defined value) and it's pointers by setting `MPDGCTRL[RST_RD_FIFO] = 1`
9. Decrement the write delay line absolute offset of each byte by 1 (i.e. `MPWRDLCTL[WR_DL_ABS_OFFSET#]` )
10. Force the delay line to measure itself and to issue the requested write delay by configuring `MPMUR[FRC_MSR] = 1`
11. Issue write command to any legal DDR address of the external DDR device
12. Issue read command, to the address written previously, from the external DDR device
13. Compare the read data byte to the associated byte in the pre-defined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it is needed to store the low write boundary of the associated byte of each byte at `MPWRDLHWST0/1[HW_WR_DL_LOW#]` . If the comparison passes then repeat steps 8-12. If all data comparisons fail then advance to the next step.
14. Start seeking the upper boundary and set the write delay line absolute offset of each byte to the initial value + 1
15. Force the delay line to measure itself and to issue the requested write delay by configuring `MPMUR[FRC_MSR] = 1`



16. Reset the rd fifo (to the inverted pre-defined value) and it's pointers by setting `MPDGCTRL[RST_RD_FIFO] = 1`
17. Issue write command to any legal DDR address of the external DDR device
18. Issue read command, to the address written previously, from the external DDR device
19. Compare the read data byte to the associated byte in the pre-defined value for all the bytes in the DDR burst (burst length 4 or 8). If the comparison fails then it is needed to store the upper write boundary of the associated byte of each byte at `MPWRDLHWST0/1[HW_WR_DL_UP#]`. If the comparison passes then increment the write delay line absolute offset of each byte by 1.
20. Force the delay line to measure itself and to issue the requested write delay by configuring `MPMUR[FRC_MSR] = 1`
21. If all read data comparisons fail then advance to the next step else repeat steps 16-20.
22. After finding the window boundary (lower and upper) of each write data byte then calculate the average between lower and upper boundaries and store the associated average at `MPWRDLCTL[WR_DL_ABS_OFFSET#]`
23. Force the delay line to measure itself and to issue the requested write delay by configuring `MPMUR[FRC_MSR] = 1`

### 32.11.6 Write leveling Calibration

The write leveling calibration can generate a delay between the clock and the associate DQS of up to 3 cycles as following:  $(WL\_DL\_ABS\_OFFSET/256 * cycle) + (WL\_HC\_DEL * half\ cycle) + (WL\_CYC\_DEL * cycle)$ .

Write leveling calibration can be executed automatically(HW) or manually (SW).

The automatic calibration process can only detect the optimal DQS to clock delay to within 1 cycle. In extreme cases in which the DDR3 memory is placed far from the microcontroller (long address/command/clock trace lengths), the skew between the DQS and clock may exceed 1 cycle. If this is the case, it is the user's responsibility to both understand that their design causes the DQS to clock skew to exceed 1 cycle and to indicate this manually in the `MPWLDECTRL0/1[WL_CYC_DEL#]`. It is highly recommended to keep the DDR3 memory as close to the microcontroller as possible, especially in embedded system designs. When using fly-by topology, the user should calculate the PCB flight time of the clock signal to the furthest placed DDR3 memory to ensure less than 1 cycle skew between DQS and clock.

#### NOTE

In LPDDR2 mode Write-leveling calibration should be disabled.

In DDR3\_x64 mode activation of the calibration is done by setting MPWLGCR[HW\_WL\_EN] at address 0xBASE1\_0808

### NOTE

It is essential to route the first bit in each data byte group (D0, D8, D16, D24, D32, D40, D48, D56) from the DDR3 memory to the same data bus bits on the controller. The DDR3 memory outputs the state of the DRAM clock during the write leveling calibration. If any of these are not routed properly, the controller will have no information regarding the state of the DRAM clock during calibration. In previous designs in which write leveling calibration was not performed, the board designer would often swap data bits within each byte group to make the data bus routing cleaner and less susceptible to noise and impedance mismatch. This can still be done with DDR3 so long as the requirement of properly routing D0, D8, D16, D24, D32, D40, D48, D56 is maintained.

### 32.11.6.1 Hardware Write Leveling Calibration

The following steps should be executed:

1. Configure the external DDR device to enter write leveling mode through MRS command
2. Activate the DQS output enable by setting MDSCR[WL\_EN]
3. Active automatic calibration by setting MPWLGCR[HW\_WL\_EN]

The following steps will be executed automatically by the MMDC:

4. MMDC enters write leveling mode, counts 25 + 15 cycles and drives the DQS pads as output while the DQ pads will remain inputs. In parallel the MMDC configures the write leveling delay line to "0" (i.e. MPWLDECTRL0[WL\_DL\_ABS\_OFFSET#] = 0) and issue measurement process of the writ-leveling delay-line to update itself with the new value
5. MMDC drives one DQS pulse to the DDR external device
6. MMDC waits 16 cycles (to guarantee that the DQ prime data is stable) and samples the associated prime DQ bit (for example for DQS1 the MMDC samples DQ[8])
7. MMDC increments the write leveling delay line by 1/8 cycle and perform measurement process in order to load the updated value to the associated delay-line
8. MMDC repeats steps 5-7 till the write leveling delay is 1 cycle

9. MMDC checks the 8 bit prime DQ results for each DQS and finds the first transition from 0 to 1. If no transition is found then the MMDC indicates an error at MPWLGCR[HW\_WL\_ERR#]
10. MMDC stores the value that issues the last "0" on the prime DQ before the transition and loads it to the write leveling delay-line. The MMDC initiates a fine-tune process by incrementing the delay-line values by 1 step (which is 1/256 part of a cycle) till detecting the most accurate transition from 0 to 1
11. Upon completion of this process the MMDC de-asserts the MPWLGCR[HW\_WL\_EN] and update the most accurate value of the delay-line at the associated MPWLDECTRL#[WL\_DL\_ABS\_OFFSET#]
12. MMDC perform measurement process in order to load the most accurate value to the associated delay-line
13. User should issue MRS command to exit write leveling mode
14. The user should read the results of the associated delay-line at MPWLDECTRL#[WL\_DL\_ABS\_OFFSET#] and in case the user estimates that the reasonable delay may be above 1 cycle then the user should indicate it at MPWLDECTRL#[WL\_CYC\_DEL#]. Moreover the user should indicate it in MDMISC[WALAT] field. For example, if the result of the write leveling calibration is 100/256 parts of a cycle, but the user estimates that the delay is above 2 cycles then MPWLDECTRL#[WL\_CYC\_DEL#] should be configured to 2, so the total delay will be 2 and 100/256 parts of a cycle
15. Return the DQS output enable to functional mode by deasserting MDSCR[WL\_EN]

### 32.11.6.2 SW Write Leveling Calibration

The following steps should be executed:

#### NOTE

It is recommended to perform the write calibration using the HW method. The SW method is provided for debug purposes only.

1. Configure the external DDR device to enter write leveling mode through MRS command
2. Activate the DQS output enable by setting MDSCR[WL\_EN]
3. Set the write-leveling delay-line offset to "0" by configuring MPWLDECTRL0[WL\_DL\_ABS\_OFFSET#] = 0
4. Force the delay line to measure itself and to issue the requested write-leveling delay by configuring MPMUR[FRC\_MSR] = 1
5. Activate SW write-leveling calibration and issue one DQS pulse by setting MPWLGCR[SW\_WL\_EN] = 1 together with MPWLGCR[SW\_WL\_CNT\_EN] = 1

6. Issue an IP read command from MPWLGCR. If MPWLGCR[SW\_WL\_EN] = 0 then the SW write-leveling result is valid at MPWLGCR[WL\_SW\_RES#].
7. Increment the write leveling delay line by 1/8 cycle (i.e add 0x20 to {MPWLDECTRL0[WL\_HC\_DEL#],MPWLDECTRL0[WL\_DL\_ABS\_OFFSET#]})
8. Force the delay line to measure itself and to issue the requested write-leveling delay by configuring MPMUR[FRC\_MSR] = 1
9. Activate SW write-leveling calibration and issue one DQS pulse by setting MPWLGCR[SW\_WL\_EN] = 1
10. Repeat steps 6-9 till the edge of CK was detected (i.e the write-leveling result switched from "0" to "1")
11. Store the value that issues the last "0" on the prime DQ before the transition and load it to the write leveling delay-line and start fine tuning process to detect the exact switch from "0" to "1"
12. Force the delay line to measure itself and to issue the requested write-leveling delay by configuring MPMUR[FRC\_MSR] = 1
13. Activate SW write-leveling calibration and issue one DQS pulse by setting MPWLGCR[SW\_WL\_EN] = 1
14. Issue an IP read command from MPWLGCR. If MPWLGCR[SW\_WL\_EN] = 0 then the SW write-leveling result is valid at MPWLGCR[WL\_SW\_RES#].
15. Increment the write leveling delay line by 1 step (i.e add 0x01 to MPWLDECTRL0[WL\_DL\_ABS\_OFFSET#])
16. Force the delay line to measure itself and to issue the requested write-leveling delay by configuring MPMUR[FRC\_MSR] = 1
17. Issue an IP read command from MPWLGCR. If MPWLGCR[SW\_WL\_EN] = 0 then the SW write-leveling result is valid at MPWLGCR[WL\_SW\_RES#].
18. Activate SW write-leveling calibration and issue one DQS pulse by setting MPWLGCR[SW\_WL\_EN] = 1
19. Repeat steps 15-18 till the exact edge of CK was detected (i.e the write-leveling result switched from "0" to "1")
20. Issue MRS command to exit write leveling mode
21. Return the DQS output enable to functional mode by deasserting MDSCR[WL\_EN]

### 32.11.7 Write fine tuning

Write fine tuning is an additional circuit that provides the ability to fine tune the timing of each dq/dm bits (relative to dqs) by up to +/-100 ps.

This is done by reducing the delay between the `wl_dqs` by 100 ps and adding a configurable delay of up to 200 ps (6 delay units of around 30-35 ps each) for each DQ/DM output. The delay can be configured independently for each DQ/DM. The calibration of this mechanism can be done only by writing & reading data from the memory. Controlled by register `MPWRDQBY#DL`.

### 32.11.8 Read fine tuning

Read fine tuning is an additional circuit that provides the ability to fine tune the timing of each coming dq bits (relative to coming dqs) by up to +/-100 ps.

This is done by reducing the delay between the incoming `rd_dqs` by 100 ps and adding a configurable delay of up to 200 ps (6 delay units of around 30-35 ps each) for each DQ input. The delay can be configured independently for each DQ. The calibration of this mechanism can be done only by writing & reading data from the memory. Controlled by register `MPRDDQBY#DL`.

## 32.12 MMDC Memory Map/Register Definition

MMDC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
21B_0000	MMDC Core Control Register (MMDC_MDCTL)	32	R/W	0311_0000h	<a href="#">32.12.1/2053</a>
21B_0004	MMDC Core Power Down Control Register (MMDC_MDPDC)	32	R/W	0003_0012h	<a href="#">32.12.2/2055</a>
21B_0008	MMDC Core ODT Timing Control Register (MMDC_MDOTC)	32	R/W	1227_2000h	<a href="#">32.12.3/2057</a>
21B_000C	MMDC Core Timing Configuration Register 0 (MMDC_MDCFG0)	32	R/W	3236_22D3h	<a href="#">32.12.4/2059</a>
21B_0010	MMDC Core Timing Configuration Register 1 (MMDC_MDCFG1)	32	R/W	B6B1_8A23h	<a href="#">32.12.5/2061</a>
21B_0014	MMDC Core Timing Configuration Register 2 (MMDC_MDCFG2)	32	R/W	00C7_0092h	<a href="#">32.12.6/2063</a>
21B_0018	MMDC Core Miscellaneous Register (MMDC_MDMISC)	32	R/W	0000_1600h	<a href="#">32.12.7/2065</a>
21B_001C	MMDC Core Special Command Register (MMDC_MDSCR)	32	R/W	0000_0000h	<a href="#">32.12.8/2068</a>
21B_0020	MMDC Core Refresh Control Register (MMDC_MDREF)	32	R/W	0000_C000h	<a href="#">32.12.9/2071</a>

Table continues on the next page...

## MMDC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
21B_002C	MMDC Core Read/Write Command Delay Register (MMDC_MDRWD)	32	R/W	0F9F_26D2h	<a href="#">32.12.10/2073</a>
21B_0030	MMDC Core Out of Reset Delays Register (MMDC_MDOR)	32	R/W	009F_0E0Eh	<a href="#">32.12.11/2075</a>
21B_0034	MMDC Core MRR Data Register (MMDC_MDMRR)	32	R	0000_0000h	<a href="#">32.12.12/2076</a>
21B_0038	MMDC Core Timing Configuration Register 3 (MMDC_MDCFG3LP)	32	R/W	0000_0000h	<a href="#">32.12.13/2077</a>
21B_003C	MMDC Core MR4 Derating Register (MMDC_MDMR4)	32	R/W	0000_0000h	<a href="#">32.12.14/2079</a>
21B_0040	MMDC Core Address Space Partition Register (MMDC_MDASP)	32	R/W	0000_003Fh	<a href="#">32.12.15/2081</a>
21B_0400	MMDC Core AXI Reordering Control Register (MMDC_MAARCR)	32	R/W	5142_01F0h	<a href="#">32.12.16/2082</a>
21B_0404	MMDC Core Power Saving Control and Status Register (MMDC_MAPSR)	32	R/W	0000_1007h	<a href="#">32.12.17/2084</a>
21B_0408	MMDC Core Exclusive ID Monitor Register0 (MMDC_MAEXIDR0)	32	R/W	0020_0000h	<a href="#">32.12.18/2086</a>
21B_040C	MMDC Core Exclusive ID Monitor Register1 (MMDC_MAEXIDR1)	32	R/W	0060_0040h	<a href="#">32.12.19/2087</a>
21B_0410	MMDC Core Debug and Profiling Control Register 0 (MMDC_MADPCR0)	32	R/W	0000_0000h	<a href="#">32.12.20/2088</a>
21B_0414	MMDC Core Debug and Profiling Control Register 1 (MMDC_MADPCR1)	32	R/W	0000_0000h	<a href="#">32.12.21/2089</a>
21B_0418	MMDC Core Debug and Profiling Status Register 0 (MMDC_MADPSR0)	32	R	0000_0000h	<a href="#">32.12.22/2090</a>
21B_041C	MMDC Core Debug and Profiling Status Register 1 (MMDC_MADPSR1)	32	R	0000_0000h	<a href="#">32.12.23/2090</a>
21B_0420	MMDC Core Debug and Profiling Status Register 2 (MMDC_MADPSR2)	32	R	0000_0000h	<a href="#">32.12.24/2091</a>
21B_0424	MMDC Core Debug and Profiling Status Register 3 (MMDC_MADPSR3)	32	R	0000_0000h	<a href="#">32.12.25/2091</a>
21B_0428	MMDC Core Debug and Profiling Status Register 4 (MMDC_MADPSR4)	32	R	0000_0000h	<a href="#">32.12.26/2092</a>
21B_042C	MMDC Core Debug and Profiling Status Register 5 (MMDC_MADPSR5)	32	R	0000_0000h	<a href="#">32.12.27/2092</a>
21B_0430	MMDC Core Step By Step Address Register (MMDC_MASBS0)	32	R	0000_0000h	<a href="#">32.12.28/2093</a>
21B_0434	MMDC Core Step By Step Address Attributes Register (MMDC_MASBS1)	32	R	0000_0000h	<a href="#">32.12.29/2093</a>
21B_0440	MMDC Core General Purpose Register (MMDC_MAGENP)	32	R/W	0000_0000h	<a href="#">32.12.30/2094</a>
21B_0800	MMDC PHY ZQ HW control register (MMDC_MPZQHWCTRL)	32	R/W	A138_0000h	<a href="#">32.12.31/2095</a>

Table continues on the next page...

**MMDC memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/ page</b>
21B_0804	MMDC PHY ZQ SW control register (MMDC_MPZQSWCTRL)	32	R/W	0000_0000h	<a href="#">32.12.32/2098</a>
21B_0808	MMDC PHY Write Leveling Configuration and Error Status Register (MMDC_MPWLGCER)	32	R/W	0000_0000h	<a href="#">32.12.33/2100</a>
21B_080C	MMDC PHY Write Leveling Delay Control Register 0 (MMDC_MPWLDECTRL0)	32	R/W	0000_0000h	<a href="#">32.12.34/2103</a>
21B_0810	MMDC PHY Write Leveling Delay Control Register 1 (MMDC_MPWLDECTRL1)	32	R/W	0000_0000h	<a href="#">32.12.35/2105</a>
21B_0814	MMDC PHY Write Leveling delay-line Status Register (MMDC_MPWLDLST)	32	R	0000_0000h	<a href="#">32.12.36/2108</a>
21B_0818	MMDC PHY ODT control register (MMDC_MPODTCTRL)	32	R/W	0000_0000h	<a href="#">32.12.37/2110</a>
21B_081C	MMDC PHY Read DQ Byte0 Delay Register (MMDC_MPRDDQBY0DL)	32	R/W	0000_0000h	<a href="#">32.12.38/2112</a>
21B_0820	MMDC PHY Read DQ Byte1 Delay Register (MMDC_MPRDDQBY1DL)	32	R/W	0000_0000h	<a href="#">32.12.39/2115</a>
21B_0824	MMDC PHY Read DQ Byte2 Delay Register (MMDC_MPRDDQBY2DL)	32	R/W	0000_0000h	<a href="#">32.12.40/2118</a>
21B_0828	MMDC PHY Read DQ Byte3 Delay Register (MMDC_MPRDDQBY3DL)	32	R/W	0000_0000h	<a href="#">32.12.41/2120</a>
21B_082C	MMDC PHY Write DQ Byte0 Delay Register (MMDC_MPWRDQBY0DL)	32	R/W	0000_0000h	<a href="#">32.12.42/2123</a>
21B_0830	MMDC PHY Write DQ Byte1 Delay Register (MMDC_MPWRDQBY1DL)	32	R/W	0000_0000h	<a href="#">32.12.43/2125</a>
21B_0834	MMDC PHY Write DQ Byte2 Delay Register (MMDC_MPWRDQBY2DL)	32	R/W	0000_0000h	<a href="#">32.12.44/2127</a>
21B_0838	MMDC PHY Write DQ Byte3 Delay Register (MMDC_MPWRDQBY3DL)	32	R/W	0000_0000h	<a href="#">32.12.45/2130</a>
21B_083C	MMDC PHY Read DQS Gating Control Register 0 (MMDC_MPDGCTRL0)	32	R/W	0000_0000h	<a href="#">32.12.46/2132</a>
21B_0840	MMDC PHY Read DQS Gating Control Register 1 (MMDC_MPDGCTRL1)	32	R/W	0000_0000h	<a href="#">32.12.47/2134</a>
21B_0844	MMDC PHY Read DQS Gating delay-line Status Register (MMDC_MPDGDLST0)	32	R	0000_0000h	<a href="#">32.12.48/2137</a>
21B_0848	MMDC PHY Read delay-lines Configuration Register (MMDC_MPRDDLCTL)	32	R/W	4040_4040h	<a href="#">32.12.49/2138</a>
21B_084C	MMDC PHY Read delay-lines Status Register (MMDC_MPRDDLST)	32	R	0000_0000h	<a href="#">32.12.50/2140</a>
21B_0850	MMDC PHY Write delay-lines Configuration Register (MMDC_MPWRDLCTL)	32	R/W	4040_4040h	<a href="#">32.12.51/2141</a>
21B_0854	MMDC PHY Write delay-lines Status Register (MMDC_MPWRDLST)	32	R	0000_0000h	<a href="#">32.12.52/2142</a>
21B_0858	MMDC PHY CK Control Register (MMDC_MPSCDCTRL)	32	R/W	0000_0000h	<a href="#">32.12.53/2143</a>

*Table continues on the next page...*

**MMDC memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/ page</b>
21B_085C	MMDC ZQ LPDDR2 HW Control Register (MMDC_MPZQLP2CTL)	32	R/W	1B5F_0109h	<a href="#">32.12.54/ 2144</a>
21B_0860	MMDC PHY Read Delay HW Calibration Control Register (MMDC_MPRDDLHWCTL)	32	R/W	0000_0000h	<a href="#">32.12.55/ 2146</a>
21B_0864	MMDC PHY Write Delay HW Calibration Control Register (MMDC_MPWRDLHWCTL)	32	R/W	0000_0000h	<a href="#">32.12.56/ 2148</a>
21B_0868	MMDC PHY Read Delay HW Calibration Status Register 0 (MMDC_MPRDDLHWST0)	32	R	0000_0000h	<a href="#">32.12.57/ 2149</a>
21B_086C	MMDC PHY Read Delay HW Calibration Status Register 1 (MMDC_MPRDDLHWST1)	32	R	0000_0000h	<a href="#">32.12.58/ 2150</a>
21B_0870	MMDC PHY Write Delay HW Calibration Status Register 0 (MMDC_MPWRDLHWST0)	32	R	0000_0000h	<a href="#">32.12.59/ 2151</a>
21B_0874	MMDC PHY Write Delay HW Calibration Status Register 1 (MMDC_MPWRDLHWST1)	32	R	0000_0000h	<a href="#">32.12.60/ 2152</a>
21B_0878	MMDC PHY Write Leveling HW Error Register (MMDC_MPWLHWERR)	32	R/W	0000_0000h	<a href="#">32.12.61/ 2153</a>
21B_087C	MMDC PHY Read DQS Gating HW Status Register 0 (MMDC_MPDGHWST0)	32	R	0000_0000h	<a href="#">32.12.62/ 2153</a>
21B_0880	MMDC PHY Read DQS Gating HW Status Register 1 (MMDC_MPDGHWST1)	32	R	0000_0000h	<a href="#">32.12.63/ 2154</a>
21B_0884	MMDC PHY Read DQS Gating HW Status Register 2 (MMDC_MPDGHWST2)	32	R	0000_0000h	<a href="#">32.12.64/ 2154</a>
21B_0888	MMDC PHY Read DQS Gating HW Status Register 3 (MMDC_MPDGHWST3)	32	R	0000_0000h	<a href="#">32.12.65/ 2155</a>
21B_088C	MMDC PHY Pre-defined Compare Register 1 (MMDC_MPPDCMPR1)	32	R/W	0000_0000h	<a href="#">32.12.66/ 2156</a>
21B_0890	MMDC PHY Pre-defined Compare and CA delay-line Configuration Register (MMDC_MPPDCMPR2)	32	R/W	0040_0000h	<a href="#">32.12.67/ 2157</a>
21B_0894	MMDC PHY SW Dummy Access Register (MMDC_MPSWDAR0)	32	R/W	0000_0000h	<a href="#">32.12.68/ 2159</a>
21B_0898	MMDC PHY SW Dummy Read Data Register 0 (MMDC_MPSWDRDR0)	32	R	FFFF_FFFFh	<a href="#">32.12.69/ 2160</a>
21B_089C	MMDC PHY SW Dummy Read Data Register 1 (MMDC_MPSWDRDR1)	32	R	FFFF_FFFFh	<a href="#">32.12.70/ 2161</a>
21B_08A0	MMDC PHY SW Dummy Read Data Register 2 (MMDC_MPSWDRDR2)	32	R	FFFF_FFFFh	<a href="#">32.12.71/ 2161</a>
21B_08A4	MMDC PHY SW Dummy Read Data Register 3 (MMDC_MPSWDRDR3)	32	R	FFFF_FFFFh	<a href="#">32.12.72/ 2161</a>
21B_08A8	MMDC PHY SW Dummy Read Data Register 4 (MMDC_MPSWDRDR4)	32	R	FFFF_FFFFh	<a href="#">32.12.73/ 2162</a>
21B_08AC	MMDC PHY SW Dummy Read Data Register 5 (MMDC_MPSWDRDR5)	32	R	FFFF_FFFFh	<a href="#">32.12.74/ 2162</a>
21B_08B0	MMDC PHY SW Dummy Read Data Register 6 (MMDC_MPSWDRDR6)	32	R	FFFF_FFFFh	<a href="#">32.12.75/ 2163</a>

*Table continues on the next page...*



**MMDC memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
21B_08B4	MMDC PHY SW Dummy Read Data Register 7 (MMDC_MPSWDRDR7)	32	R	FFFF_FFFFh	<a href="#">32.12.76/2163</a>
21B_08B8	MMDC PHY Measure Unit Register (MMDC_MPMUR0)	32	R/W	0000_0000h	<a href="#">32.12.77/2164</a>
21B_08BC	MMDC Write CA delay-line controller (MMDC_MPWRCADL)	32	R/W	0000_0000h	<a href="#">32.12.78/2165</a>
21B_08C0	MMDC Duty Cycle Control Register (MMDC_MPDCCR)	32	R	2492_2492h	<a href="#">32.12.79/2167</a>

**32.12.1 MMDC Core Control Register (MMDC\_MDCTL)**

Address: 21B\_0000h base + 0h offset = 21B\_0000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R				0					0					0		
W	SDE_0	SDE_1											BL			DSIZ
Reset	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R									0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MDCTL field descriptions**

Field	Description
31 SDE_0	MMDC Enable CS0. This bit enables/disables accesses from the MMDC toward Chip Select 0. The reset value of this bit is "0" (i.e No clocks and clock enable will be driven to the memory).  At the enabling point the MMDC will perform an initialization process (including a delay on RESET and/or CKE) for both chip selects. The initialization length depends on the configured memory type.  0 Disabled 1 Enabled
30 SDE_1	MMDC Enable CS1. This bit enables/disables accesses from the MMDC toward Chip Select 1. The reset value of this bit is "0" (i.e No clocks and clock enable will be driven to the memory).  At the enabling point the MMDC will perform an initialization process (including a delay on RESET and/or CKE) for both chip selects. The initialization length depends on the configured memory type.  0 Disabled 1 Enabled

*Table continues on the next page...*

### MMDC\_MDCTL field descriptions (continued)

Field	Description
29–27 Reserved	This read-only field is reserved and always has the value 0.
26–24 ROW	<p>Row Address Width. This field specifies the number of row addresses used by the memory array. It will affect the way an incoming address will be decoded.</p> <p>Settings 110-111 are reserved</p> <p>000 11 bits Row  001 12 bits Row  010 13 bits Row  011 14 bits Row  100 15 bits Row  101 16 bits Row</p>
23 Reserved	This read-only field is reserved and always has the value 0.
22–20 COL	<p>Column Address Width. This field specifies the number of column addresses used by the memory array. It will determine how an incoming address will be decoded.</p> <p>0x0 9 bits column  0x1 10 bits column  0x2 11 bits column  0x3 8 bits column  0x4 12 bits column  0x5-0xF Reserved</p>
19 BL	<p>Burst Length. This field determines the burst length of the DDR device.</p> <p>In LPDDR2 mode the MMDC supports burst length 4.</p> <p>In DDR3 mode the MMDC supports burst length 8.</p> <p>0 Burst Length 4 is used  1 Burst Length 8 is used</p>
18 Reserved	This read-only field is reserved and always has the value 0.
17–16 DSIZ	<p>DDR data bus size. This field determines the size of the data bus of the DDR memory</p> <p>0 16-bit data bus  1 32-bit data bus  —  —  2-3 Reserved</p>
15–0 Reserved	This read-only field is reserved and always has the value 0.

## 32.12.2 MMDC Core Power Down Control Register (MMDC\_MDPDC)

Table 32-13. PRCT field encoding

PRCT[2:0]	Precharge Timer
000	Disabled (Bit field reset value)
001	2 clocks
010	4 clocks
011	8 clocks
100	16 clocks
101	32 clocks
110	64 clocks
111	128 clocks

Table 32-14. PWDT field encoding

PWDT[3:0]	Power Down Time-out
0000	Disabled (bit field reset value)
0001	16 cycles
0010	32 cycles
0011	64 cycles
0100	128 cycles
0101	256 cycles
0110	512 cycles
0111	1024 cycles
1000	2048 cycles
1001	4096 cycles
1010	8196 cycles
1011	16384 cycles
1100	32768 cycles
1101-1111	Reserved

Address: 21B\_0000h base + 4h offset = 21B\_0004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	PRCT_1				0	PRCT_0				0				tCKE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1

## MMDC Memory Map/Register Definition

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PWDT_1				PWDT_0				SLOW_PD	BOTH_CS_PD	tCKSRX			tCKSRE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	1	0

### MMDC\_MDPDC field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–28 PRCT_1	Precharge Timer - Chip Select 1. This field determines the amount of idle cycle for which chip select 1 will be automatically precharged. The amount of cycles are determined according to the PRCT Field Encoding table above.
27 Reserved	This read-only field is reserved and always has the value 0.
26–24 PRCT_0	Precharge Timer - Chip Select 0. This field determines the amount of idle cycle for which chip select 0 will be automatically precharged. The amount of cycles are determined according to the table below.
23–19 Reserved	This read-only field is reserved and always has the value 0.
18–16 tCKE	CKE minimum pulse width. This field determines the minimum pulse width of CKE.  0x0 1 cycle 0x1 2 cycles 0x6 7 cycles 0x7 8 cycles
15–12 PWDT_1	Power Down Timer - Chip Select 1. This field determines the amount of idle cycle for which chip select 1 will be automatically get into precharge/active power down. The amount of cycles are determined according to the PWDT Field Encoding table above.
11–8 PWDT_0	Power Down Timer - Chip Select 0. This field determines the amount of idle cycle for which chip select 0 will be automatically get into precharge/active power down. The amount of cycles are determined according to the PWDT Field Encoding table above.
7 SLOW_PD	Slow/fast power down. In DDR3 mode this field is referred to slow precharge power-down. In LPDDR2 mode this field is not relevant.  <b>NOTE:</b> Memory should be configured the same.  0 Fast mode. 1 Slow mode.
6 BOTH_CS_PD	Parallel power down entry to both chip selects. When power down timer is used for both chip-selects (i.e PWDT_0 and PWDT1 don't equal "0" ), then if this bit is enabled, the MMDC will enter power down only if the amount of idle cycles of both chip selects was obtained.

Table continues on the next page...

## MMDC\_MDPDC field descriptions (continued)

Field	Description
	0 Each chip select can enter power down independently according to its configuration. 1 Chip selects can enter power down only if the amount of idle cycles of both chip selects was obtained.
5–3 tCKSRX	Valid clock cycles before self-refresh exit. This field determines the amount of clock cycles before self-refresh exit  0x0 0 cycle 0x1 1 cycles 0x6 6 cycles 0x7 7 cycles
2–0 tCKSRE	Valid clock cycles after self-refresh entry. This field determines the amount of clock cycles after self-refresh entry  0x0 0 cycle 0x1 1 cycles 0x6 6cycles 0x7 7cycles

### 32.12.3 MMDC Core ODT Timing Control Register (MMDC\_MDOTC)

For further information see [ODT Configuration](#).

Address: 21B\_0000h base + 8h offset = 21B\_0008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	1	0	0	1	0	0	0	1	0	0	1	1	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W																
Reset	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0

## MMDC\_MDOTC field descriptions

Field	Description
31–30 Reserved	This read-only field is reserved and always has the value 0.
29–27 tAOFPD	Asynchronous RTT turn-off delay (power down with DLL frozen). This field determines the time between termination circuit starts to turn off the ODT resistance till termination has reached high impedance.  This field is not relevant in LPDDR2 mode.  0x0 1 cycle

Table continues on the next page...

### MMDC\_MDOTC field descriptions (continued)

Field	Description
	0x1 2 cycles 0x6 7 cycles 0x7 8 cycles
26–24 tAONPD	Asynchronous RTT turn-on delay (power down with DLL frozen). This field determines the time between termination circuit gets out of high impedance and begins to turn on till ODT resistance are fully on. This field is not relevant in LPDDR2 mode. 0x0 1 cycle 0x1 2 cycles 0x6 7 cycles 0x7 8 cycles
23–20 tANPD	Asynchronous ODT to power down entry delay. In DDR3 should be set to tCWL-1 This field is not relevant in LPDDR2 mode. 0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0xE 15 clocks 0xF 16 clocks
19–16 tAXPD	Asynchronous ODT to power down exit delay. In DDR3 should be set to tCWL-1 This field is not relevant in LPDDR2 mode. 0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0xE 15 clocks 0xF 16 clocks
15 Reserved	This read-only field is reserved and always has the value 0.
14–12 tODTLon	ODT turn on latency. This field determines the delay between ODT signal and the associated RTT, where according to JEDEC standard it equals WL(write latency) - 2. Therefore, the value that is configured to tODTLon field should correspond the value that is configured to MDCGFG1[tCWL] In LPDDR2 this field is not relevant. 0x0 - 0x1 Reserved 0x2 2 cycles 0x3 3 cycles 0x4 4 cycles 0x5 5 cycles 0x6 6 cycles 0x7 Reserved
11–9 Reserved	This read-only field is reserved and always has the value 0.
8–4 tODT_idle_off	ODT turn off latency. This field determines the Idle period before turning memory ODT off. This field is not relevant in LPDDR2 mode. 0x0 0 cycle (turned off at the earliest possible time)

Table continues on the next page...

**MMDC\_MDOTC field descriptions (continued)**

Field	Description
	0x1 1 cycle 0x2 2 cycles 0x1E 30 cycles 0x1F 31 cycles
3–0 Reserved	This read-only field is reserved and always has the value 0.

## 32.12.4 MMDC Core Timing Configuration Register 0 (MMDC\_MDCFG0)

Address: 21B\_0000h base + Ch offset = 21B\_000Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	tRFC								tXS								tXP			tXPDLL			tFAW				tCL					
W																																
Reset	0	0	1	1	0	0	1	0	0	0	1	1	0	1	1	0	0	0	1	0	0	0	1	0	1	1	0	1	0	0	1	1

**MMDC\_MDCFG0 field descriptions**

Field	Description
31–24 tRFC	Refresh command to Active or Refresh command time.  See DDR3 SDRAM Specification JESD79-3E (July 2010) and LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter.  0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0xFE 255 clocks 0xFF 256 clocks
23–16 tXS	Exit self refresh to non READ command. In LPDDR2 it is called tXSR, self-refresh exit to next valid command delay.  See DDR3 SDRAM Specification JESD79-3E (July 2010) and LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter.  0x0 - 0x15 reserved 0x16 23 clocks 0x17 24 clocks 0xFE 255 clocks 0xFF 256 clocks
15–13 tXP	Exit power down with DLL-on to any valid command. Exit power down with DLL-frozen to commands not requiring a locked DLL.  In LPDDR2 mode this field is referred to Exit power-down to next valid command delay.  See DDR3 SDRAM Specification JESD79-3E (July 2010) and LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter.

*Table continues on the next page...*

### MMDC\_MDCFG0 field descriptions (continued)

Field	Description
	0x0 1 cycle 0x1 2 cycles 0x6 7 cycles 0x7 8 cycles
12–9 tXPDLL	Exit precharge power down with DLL frozen to commands requiring DLL. This field is not relevant in LPDDR2 mode. See DDR3 SDRAM Specification JESD79-3E (July 2010) and LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter. 0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0xE 15 clocks 0xF 16 clocks
8–4 tFAW	Four Active Window (all banks). See DDR3 SDRAM Specification JESD79-3E (July 2010) and LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter. 0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0x1E 31 clocks 0x1F 32 clocks
3–0 tCL	CAS Read Latency. In DDR3 mode this field is referred to CL. In LPDDR2 mode this field is referred to RL. <b>NOTE:</b> In LPDDR2 mode only the RL/WL pairs are allowed as specified in MR2 register See DDR3 SDRAM Specification JESD79-3E (July 2010) and LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter. 0x0 3 cycles 0x1 4 cycles 0x2 5 cycles 0x3 6 cycles 0x4 7 cycles 0x5 8 cycles 0x6 9 cycles 0x7 10 cycles 0x8 11 cycles 0x9 - 0xF Reserved



## 32.12.5 MMDC Core Timing Configuration Register 1 (MMDC\_MDCFG1)

Address: 21B\_0000h base + 10h offset = 21B\_0010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
	tRCD				tRP			tRC				tRAS				
Reset	1	0	1	1	0	1	1	0	1	0	1	1	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
	tRPA	0			tWR			tMRD				0		tCWL		
Reset	1	0	0	0	1	0	1	0	0	0	1	0	0	0	1	1

**MMDC\_MDCFG1 field descriptions**

Field	Description
31–29 tRCD	<p>Active command to internal read or write delay time (same bank).</p> <p>(This field is valid only for DDR3 memories)</p> <p>In LPDDR2 mode this parameter should be configured at tRCD_LP.</p> <p>See DDR3 SDRAM Specification JESD79-3E (July 2010) and LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter.</p> <p>0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0x3 4 clocks 0x4 5 clocks 0x5 6 clocks 0x6 7 clocks 0x7 8 clocks</p>
28–26 tRP	<p>Precharge command period (same bank).</p> <p>(This field is valid only for DDR3 memories)</p> <p>In LPDDR2 mode this parameter should be configured at tRPpb_LP.</p> <p>See DDR3 SDRAM Specification JESD79-3E (July 2010) and LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter.</p> <p>0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0x3 4 clocks 0x4 5 clocks 0x5 6 clocks 0x6 7 clocks 0x7 8 clocks</p>
25–21 tRC	<p>Active to Active or Refresh command period (same bank).</p> <p>(This field is valid only for DDR3 memories)</p>

*Table continues on the next page...*

### MMDC\_MDCFG1 field descriptions (continued)

Field	Description
	<p>In LPDDR2 mode this parameter should be configured at tRC_LP.</p> <p>See DDR3 SDRAM Specification JESD79-3E (July 2010) and LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter.</p> <p>0x0 1 clock  0x1 2 clocks  0x2 3 clocks  0x1E 31 clocks  0x1F 32 clocks</p>
20–16 tRAS	<p>Active to Precharge command period (same bank).</p> <p>See DDR3 SDRAM Specification JESD79-3E (July 2010) and LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter.</p> <p>0x0 1 clock  0x1 2 clocks  0x2 3 clocks  0x1E 31 clocks  0x1F Reserved</p>
15 tRPA	<p>Precharge-all command period.</p> <p>(This field is valid only for DDR3 memories)</p> <p>In LPDDR2 mode this parameter should be configured at tRPab_LP.</p> <p>See DDR3 SDRAM Specification JESD79-3E (July 2010) and LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter.</p> <p>0 Will be equal to: tRP.  1 Will be equal to: tRP+1.</p>
14–12 Reserved	This read-only field is reserved and always has the value 0.
11–9 tWR	<p>WRITE recovery time (same bank).</p> <p>See DDR3 SDRAM Specification JESD79-3E (July 2010) and LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter.</p> <p>0x0 1cycle  0x1 2cycles  0x2 3cycles  0x3 4cycles  0x4 5cycles  0x5 6cycles  0x6 7cycles  0x7 8 cycles</p>
8–5 tMRD	<p>Mode Register Set command cycle (all banks).</p> <p>In DDR3 mode this field should be set to max (tMRD,tMOD).</p> <p>In LPDDR2 mode this field should be set to max(tMRR,tMRW)</p> <p>See DDR3 SDRAM Specification JESD79-3E (July 2010) and LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter.</p>

Table continues on the next page...

**MMDC\_MDCFG1 field descriptions (continued)**

Field	Description
	0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0xE 15 clocks 0xF 16 clocks
4–3 Reserved	This read-only field is reserved and always has the value 0.
2–0 tCWL	CAS Write Latency. In DDR3 mode this field is referred to CWL. In LPDDR2 mode this field is referred to WL.  0x0 2cycles ( DDR3 ) , 1 cycle (LPDDR2) 0x1 3cycles ( DDR3 ) , 2 cycles (LPDDR2) 0x2 4cycles ( DDR3 ) , 3 cycles (LPDDR2) 0x3 5cycles ( DDR3 ) , 4 cycles (LPDDR2) 0x4 6cycles ( DDR3 ) , 5 cycles (LPDDR2) 0x5 7cycles ( DDR3 ) , 6 cycles (LPDDR2) 0x6 8cycles ( DDR3 ) , 7 cycles (LPDDR2) 0x7 Reserved

**32.12.6 MMDC Core Timing Configuration Register 2 (MMDC\_MDCFG2)**

Address: 21B\_0000h base + 14h offset = 21B\_0014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								tDLLK								0				tRTP				tWTR		tRRD					
W																																
Reset	0	0	0	0	0	0	0	0	1	1	0	0	0	1	1	1	0	0	0	0	0	0	0	0	1	0	0	1	0	0	1	0

**MMDC\_MDCFG2 field descriptions**

Field	Description
31–25 Reserved	This read-only field is reserved and always has the value 0.
24–16 tDLLK	DLL locking time. This field is not relevant in LPDDR2 mode. See DDR3 SDRAM Specification JESD79-3E (July 2010) and LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter.  0x0 1 cycle. 0x1 2 cycles. 0x2 3 cycles. 0xC7 200 cycles

*Table continues on the next page...*

### MMDC\_MDCFG2 field descriptions (continued)

Field	Description
	0x1FE 511 cycles. 0x1FF 512 cycles (JEDEC value for DDR3).
15–9 Reserved	This read-only field is reserved and always has the value 0.
8–6 tRTP	Internal READ command to Precharge command delay (same bank). See DDR3 SDRAM Specification JESD79-3E (July 2010) and LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter.  0x0 1cycle 0x1 2cycles 0x2 3cycles 0x3 4cycles 0x4 5cycles 0x5 6cycles 0x6 7cycles 0x7 8 cycles
5–3 tWTR	Internal WRITE to READ command delay (same bank). See DDR3 SDRAM Specification JESD79-3E (July 2010) and LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter.  0x0 1cycle 0x1 2cycles 0x2 3cycles 0x3 4cycles 0x4 5cycles 0x5 6cycles 0x6 7cycles 0x7 8 cycles
2–0 tRRD	Active to Active command period (all banks). See DDR3 SDRAM Specification JESD79-3E (July 2010) and LPDDR2 SDRAM Specification JESD209-2B (February 2010) for a detailed description of this parameter.  0x0 1cycle 0x1 2cycles 0x2 3cycles 0x3 4cycles 0x4 5cycles 0x5 6cycles 0x6 7cycles 0x7 Reserved

### 32.12.7 MMDC Core Miscellaneous Register (MMDC\_MDMISC)

Address: 21B\_0000h base + 18h offset = 21B\_0018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16						
R	CS0_RDY		CS1_RDY		0								CK1_GATING		CALIB_PER_CS		ADDR_MIRROR		LHD		WALAT	
W																						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0					

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	0			BL_ON	LPDDR2_S2	MIF3_MODE			RALAT			DDR_4_BANK	DDR_TYPE		0	RST	0
W																	
Reset	0	0	0	1	0	1	1	0	0	0	0	0	0	0	0	0	

**MMDC\_MDMISC field descriptions**

Field	Description
31 CS0_RDY	External status device on CS0. This is a read-only status bit, that indicates whether the external memory is in wake-up period.  0 Device in wake-up period. 1 Device is ready for initialization.
30 CS1_RDY	External status device on CS1. This is a read-only status bit, that indicates whether the external memory is in wake-up period.  0 Device in wake-up period. 1 Device is ready for initialization.

*Table continues on the next page...*

## MMDC\_MDMISC field descriptions (continued)

Field	Description
29–22 Reserved	This read-only field is reserved and always has the value 0.
21 CK1_GATING	Gating the secondary DDR clock. When this bit is asserted then the MMDC will disable the secondary DDR clock  0 MMDC drives two clocks toward the DDR memory 1 MMDC drives only one clock toward the DDR memory (CK0)
20 CALIB_PER_CS	Number of chip-select for calibration process. This bit determines the chip-select index that the associated calibration is targetted to. Relevant for read, write, write leveling and read DQS gating calibrations  0 Calibration is targetted to CS0 1 Calibration is targetted to CS1
19 ADDR_MIRROR	Address mirroring. <b>NOTE:</b> This feature is not supported for LPDDR2 memories. But only for DDR3 memories. For further information see <a href="#">Address mirroring</a> .  0 Address mirroring disabled. 1 Address mirroring enabled.
18 LHD	Latency hiding disable.  This is a debug feature. When set to "1" the MMDC will handle one read/write access at a time. Meaning that the MMDC pipe-line will be limited to 1 open access (next AXI address phase will be acknowledged if the current AXI data phase had finished)  0 Latency hiding on. 1 Latency hiding disable.
17–16 WALAT	Write Additional latency.  In case the write-leveling calibration process indicates a delay around half cycle (between CK and the associated DQS) then this field must be configured accordingly.  This field will add delay on the oboe I/O control, which will compensate on the additional write leveling delay on DQS and prevent the DQS from being cropped.  0x0 No additional latency required. 0x1 1 cycle additional delay 0x2 2 cycles additional delay 0x3 3 cycles additional delay
15–13 Reserved	This read-only field is reserved and always has the value 0.
12 BI_ON	Bank Interleaving On. This bit controls the organization of the bank, row and column address bits. For further information see <a href="#">Address decoding</a> .  0 Banks are not interleaved, and address will be decoded as bank-row-column 1 Banks are interleaved, and address will be decoded as row-bank-column
11 LPDDR2_S2	LPDDR2 S2 device type indication.  In case LPDDR2 device is used (DDR_TYPE = 0x1), this bit will indicate whether S2 or S4 device is used.  This bit should be cleared in DDR3 mode

*Table continues on the next page...*

**MMDC\_MDMISC field descriptions (continued)**

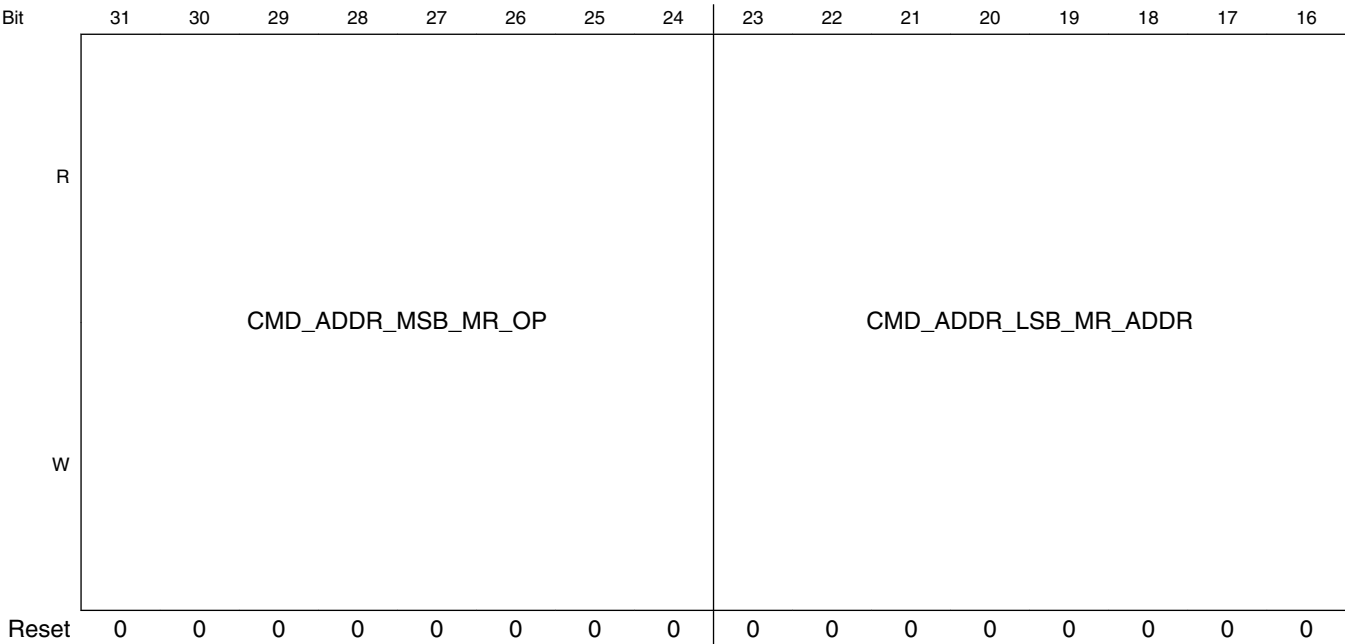
Field	Description
	0x0 LPDDR2-S4 device is used. 0x1 LPDDR2-S2 device is used.
10–9 MIF3_MODE	Command prediction working mode. This field determines the level of command prediction that will be used by the MMDC  00 Disable prediction. 01 Enable prediction based on : Valid access on first pipe line stage. 10 Enable prediction based on: Valid access on first pipe line stage, Valid access on axi bus. 11 Enable prediction based on: Valid access on first pipe line stage, Valid access on axi bus, Next miss access from access queue.
8–6 RALAT	Read Additional Latency. This field determines the additional read latency which is added to CAS latency and internal delays for which the MMDC will retrieve the read data from the internal FIFO. This field is used to compensate on board/chip delays.  <b>NOTE:</b> In LPDDR2 mode 2 extra cycles will be added internally in order to compensate tDQSCK delay.  0x0 no additional latency. 0x1 1 cycle additional latency. 0x2 2 cycles additional latency. 0x3 3 cycles additional latency. 0x4 4 cycles additional latency. 0x5 5 cycles additional latency. 0x6 6 cycles additional latency. 0x7 7 cycles additional latency.
5 DDR_4_BANK	Number of banks per DDR device. When this bit is set to "1" then the MMDC will work with DDR device of 4 banks.  0 8 banks device is being used. (Default) 1 4 banks device is being used
4–3 DDR_TYPE	DDR TYPE. This field determines the type of the external DDR device.  0x0 DDR3 device is used. (Default) 0x1 LPDDR2 device is used. — — 0x2 Reserved. 0x3 Reserved.
2 Reserved	This read-only field is reserved and always has the value 0.
1 RST	Software Reset. When this bit is asserted then the internal FSMs and registers of the MMDC will be initialized.  <b>NOTE:</b> This bit once asserted gets deasserted automatically.  0 Do nothing. 1 Assert reset to the MMDC.
0 Reserved	This read-only field is reserved and always has the value 0.

### 32.12.8 MMDC Core Special Command Register (MMDC\_MDSCR)

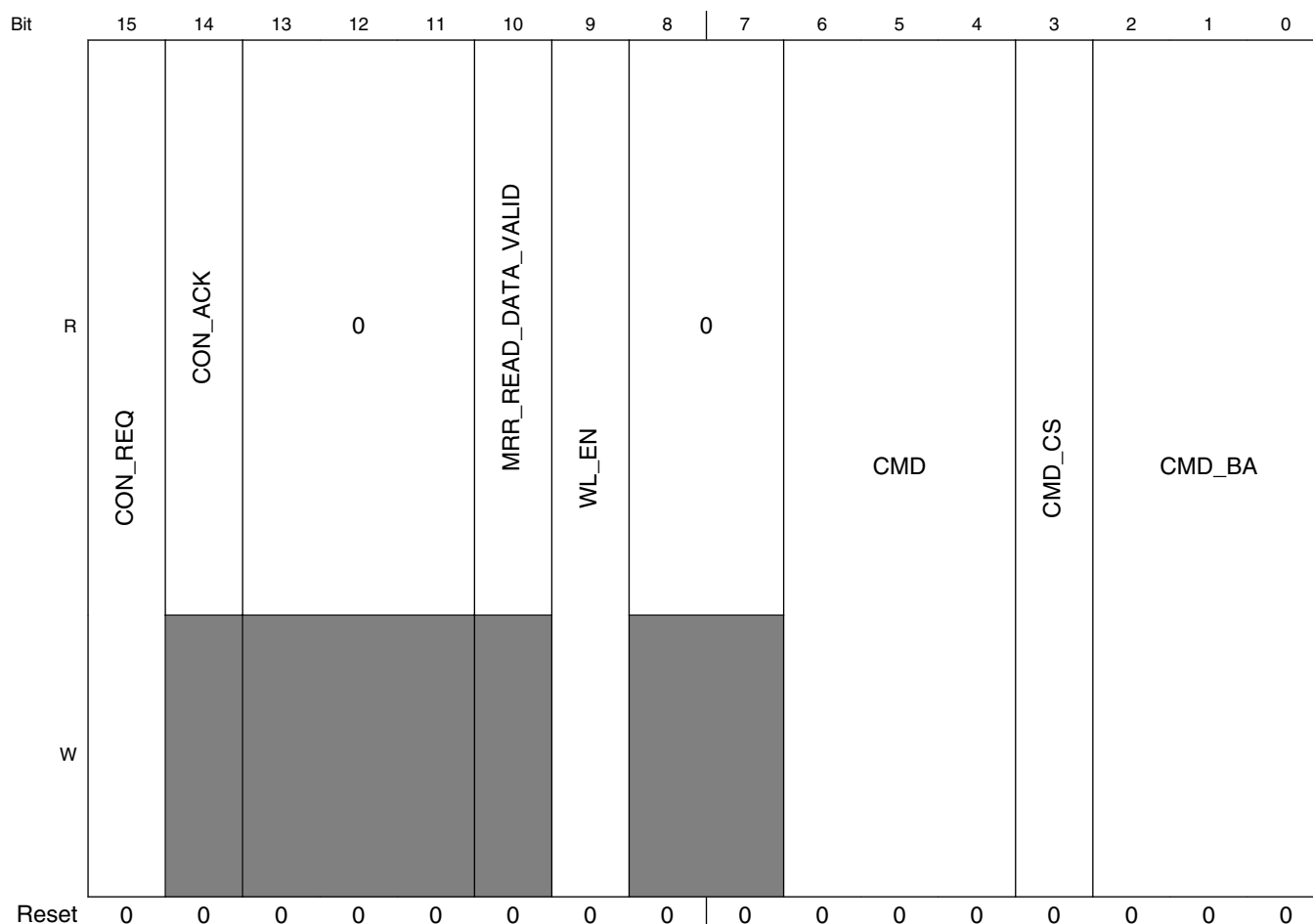
This register is used to issue special commands manually toward the external DDR device (such as load mode register, manual self refresh, manual precharge and so on). Every write to this register will be interpreted as a command, and a read from this register will show the last command that was executed.

Every write to this register will result in one special command, and the IP bus will assert `ips_xfr_wait` as long as the special command is being carried out.

Address: 21B\_0000h base + 1Ch offset = 21B\_001Ch







MMDC\_MDSCR field descriptions

Field	Description
31–24 CMD_ADDR_ MSB_MR_OP	Command/Address MSB. This field indicates the MSB of the command/Address. In LPDDR2 this field indicates the MRW operand
23–16 CMD_ADDR_ LSB_MR_ADDR	Command/Address LSB. This field indicates the LSB of the command/Address In LPDDR2 this field indicates the MRR/MRW address
15 CON_REQ	<p>Configuration request.</p> <p>When this bit is set then the MMDC will clean the pending AXI accesses and will prevent from further AXI accesses to be acknowledged. This field guarantee safe configuration (or change configuration) of the MMDC while no access is in process and prevents an unexpected behaviour.</p> <p>After setting this bit, it is needed to poll on CON_ACK until it is set to "1". When CON_ACK is asserted then configuration is permitted. After configuration is completed then this bit must be deasserted in order to process further AXI accesses.</p> <p><b>NOTE:</b> This bit is asserted at the end of the reset sequence, meaning that the MMDC is waiting to configure and initialize the external memory before accepting any AXI accesses. Configuration request/acknowledge mechanism should be used for the following procedures: changing of timing parameters , during calibration process or driving commands via MDSCR[CMD]</p>

Table continues on the next page...

### MMDC\_MDSCR field descriptions (continued)

Field	Description
	0 No request to configure MMDC. 1 A request to configure MMDC is valid
14 CON_ACK	Configuration acknowledge. Whenever this bit is set, it is permitted to configure MMDC IP registers. 0 Configuration of MMDC registers is forbidden. 1 Configuration of MMDC registers is permitted.
13–11 Reserved	This read-only field is reserved and always has the value 0.
10 MRR_READ_DATA_VALID	MRR read data valid. This field indicates that read data is valid at MDMRR register This field is relevant only for LPDDR2 mode 0 Cleared upon the assertion of MRR command 1 Set after MRR data is valid and stored at MDMRR register.
9 WL_EN	DQS pads direction. This bit controls the DQS pads direction during write-leveling calibration process. Before starting the write-leveling calibration process this bit should be set to "1". It should be set to "0" when sending write leveling exit command. For further information see <a href="#">Write leveling Calibration</a> . 0 Exit write leveling mode or stay in normal mode. 1 Write leveling entry command was sent.
8–7 Reserved	This read-only field is reserved and always has the value 0.
6–4 CMD	Command. This field contains the command to be executed. This field will be automatically cleared after the command will be send to the DDR memory. 0x0 Normal operation 0x1 Precharge all, command is sent independently of bank status (set correct CMD_CS). Will be issued even if banks are closed. Mainly used for init sequence purpose. 0x2 Auto-Refresh Command (set correct CMD_CS). 0x3 Load Mode Register Command ( DDR3, set correct CMD_CS, CMD_BA, CMD_ADDR_LSB, CMD_ADDR_MSB), MRW Command (LPDDR2, set correct CMD_CS, MR_OP, MR_ADDR) 0x4 ZQ calibration ( DDR3, set correct CMD_CS, {CMD_ADDR_MSB,CMD_ADDR_LSB} = 0x400 or 0x0 ) 0x5 Precharge all, only if banks open (set correct CMD_CS). 0x6 MRR command (LPDDR2, set correct CMD_CS, MR_ADDR) 0x7 Reserved
3 CMD_CS	Chip Select. This field determines which chip select the command is targeted to 0 to Chip-select 0 1 to Chip-select 1
2–0 CMD_BA	Bank Address. This field determines the address of the bank within the selected chip-select where the command is targetted to. 0x0 bank address 0 0x1 bank address 1

Table continues on the next page...

**MMDC\_MDSCR field descriptions (continued)**

Field	Description
0x2	bank address 2
0x7	bank address 7

**32.12.9 MMDC Core Refresh Control Register (MMDC\_MDREF)**

This register determines the refresh scheme that will be executed toward the DDR device. It specifies how often a refresh cycle occurs and how many refresh commands will be executed every refresh cycle.

For further information see [Refresh Scheme](#).

The following tables show examples of possible refresh schemes.

**Table 32-22. Refresh rate example for REF\_SEL = 0**

REFR[2:0]	Number of refresh commands every 64KHz	Average periodic refresh rate (tREFI)	System Refresh period
0x0	1	15.6 $\mu$ s	tRFC
0x1	2	7.8 $\mu$ s	2*tRFC
0x3	4	3.9 $\mu$ s	4*tRFC
0x7	8	1.95 $\mu$ s	8*tRFC

**Table 32-23. Refresh rate example for REF\_SEL = 1**

REFR[2:0]	Number of refresh commands every 32KHz	Average periodic refresh rate (tREFI)	System Refresh period
0x1	2	15.6 $\mu$ s	2*tRFC
0x3	4	7.8 $\mu$ s	4*tRFC
0x7	8	3.9 $\mu$ s	8*tRFC

**Table 32-24. Refresh rate example for REF\_SEL = 2@ 400MHz**

REFR[2:0]	Number of refresh commands every refresh cycle	REF_CNT	Average periodic refresh rate (tREFI)	System Refresh period
0x0	1	0x618	3.9 $\mu$ s	tRFC
0x1	2	0xC30	3.9 $\mu$ s	2*tRFC
0x2	3	0x1248	3.9 $\mu$ s	3*tRFC
0x3	4	0x1860	3.9 $\mu$ s	4*tRFC

Other refresh configurations are also allowed; the configuration values in the tables above are only examples for obtaining the desired average periodic refresh rate.

If the required average periodic refresh rate (tREFI) is kept, all of the rows will be refreshed in every refresh window. Because the memory device issues additional refresh commands for every refresh it receives, the tREFI remains the same across the device, regardless of its number of rows. This is particularly relevant in the tRFC parameter, which becomes bigger as the density increases.

Address: 21B\_0000h base + 20h offset = 21B\_0020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	REF_CNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REF_SEL		REFR			0										START_REF
W																
Reset	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### MMDC\_MDREF field descriptions

Field	Description
31–16 REF_CNT	Refresh Counter at DDR clock period If REF_SEL equals '2' a refresh cycle will begin every amount of DDR cycles configured in this field.  0x0     Reserved. 0x1     1 cycle. 0xFFFFE 65534 cycles. 0xFFFF 65535 cycles.
15–14 REF_SEL	Refresh Selector. This bit selects the source of the clock that will trigger each refresh cycle:  0     Periodic refresh cycles will be triggered in frequency of 64KHz. 1     Periodic refresh cycles will be triggered in frequency of 32KHz. 2     Periodic refresh cycles will be triggered every amount of cycles that are configured in REF_CNT field. 3     No refresh cycles will be triggered.
13–11 REFR	Refresh Rate. This field determines how many refresh commands will be issued every refresh cycle. After every refresh command the MMDC won't drive any command to the DDR device untill satisfying tRFC period

*Table continues on the next page...*

**MMDC\_MDREF field descriptions (continued)**

Field	Description
	0x0 1 refresh 0x1 2 refreshes 0x2 3 refreshes 0x3 4 refreshes 0x4 5 refreshes 0x5 6 refreshes 0x6 7 refreshes 0x7 8 refreshes
10–1 Reserved	This read-only field is reserved and always has the value 0.
0 START_REF	Manual start of refresh cycle. When this field is set to '1' the MMDC will start a refresh cycle immediately according to number of refresh commands that are configured in 'REFR' field. This bit returns to zero automatically. 0 Do nothing. 1 Start a refresh cycle.

**32.12.10 MMDC Core Read/Write Command Delay Register (MMDC\_MDRWD)**

This register determines the delay between back to back read and write accesses. The register reset values are set to the minimum required value. As the default values are set to achieve optimal results, changing them is discouraged.

Address: 21B\_0000h base + 2Ch offset = 21B\_002Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			tDAI												
W																
Reset	0	0	0	0	1	1	1	1	1	0	0	1	1	1	1	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	RTW_SAME			WTR_DIFF			WTW_DIFF			RTW_DIFF			RTR_DIFF		
W																
Reset	0	0	1	0	0	1	1	0	1	1	0	1	0	0	1	0

**MMDC\_MDRWD field descriptions**

Field	Description
31–29 Reserved	This read-only field is reserved and always has the value 0.
28–16 tDAI	Device auto initialization period.(maximum) This field is relevant only to LPDDR2 mode

*Table continues on the next page...*

### MMDC\_MDRWD field descriptions (continued)

Field	Description
	0x0 1 cycle 0xF9F 4000 cycles (Default, JEDEC value for LPDDR2, gives 10us at 400MHz clock). 0x1FFF 8192 cycles
15 Reserved	This read-only field is reserved and always has the value 0.
14–12 RTW_SAME	Read to write delay for the same chip-select. This field controls the delay between read to write commands toward the same chip select. The total delay is calculated according to: $BL/2 + RTW\_SAME + (tCL - tCWL) + RALAT$ 0x0 0 cycle 0x1 1 cycle 0x2 2 cycles (Default) 0x3 3 cycles 0x4 4 cycles 0x5 5 cycles 0x6 6 cycles 0x7 7 cycles
11–9 WTR_DIFF	Write to read delay for different chip-select. This field controls the delay between write to read commands toward different chip select. The total delay is calculated according to: $BL/2 + WTR\_DIFF + (tCL - tCWL) + RALAT$ 0x0 0 cycle 0x1 1 cycle 0x2 2 cycles 0x3 3 cycles (Default) 0x4 4 cycles 0x5 5 cycles 0x6 6 cycles 0x7 7 cycles
8–6 WTW_DIFF	Write to write delay for different chip-select. This field controls the delay between write to write commands toward different chip select. The total delay is calculated according to: $BL/2 + WTW\_DIFF$ 0x0 0 cycle 0x1 1 cycle 0x2 2 cycles 0x3 3 cycles (Default) 0x4 4 cycles 0x5 5 cycles 0x6 6 cycles 0x7 7 cycles
5–3 RTW_DIFF	Read to write delay for different chip-select. This field controls the delay between read to write commands toward different chip select. The total delay is calculated according to: $BL/2 + RTW\_DIFF + (tCL - tCWL) + RALAT$ 0x0 0 cycle 0x1 1 cycle

Table continues on the next page...

**MMDC\_MDRWD field descriptions (continued)**

Field	Description
	0x2 2 cycles (Default) 0x3 3 cycles 0x4 4 cycles 0x5 5 cycles 0x6 6 cycles 0x7 7 cycles
2–0 RTR_DIFF	Read to read delay for different chip-select. This field controls the delay between read to read commands toward different chip select.  The total delay is calculated according to: $BL/2 + RTR\_DIFF$  0x0 0 cycle 0x1 1 cycle 0x2 2 cycles (Default) 0x3 3 cycles 0x4 4 cycles 0x5 5 cycles 0x6 6 cycles 0x7 7 cycles

### 32.12.11 MMDC Core Out of Reset Delays Register (MMDC\_MDOR)

This register defines delays that must be kept when MMDC exits reset.

Address: 21B\_0000h base + 30h offset = 21B\_0030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	0								tXPR								0		SDE_to_RST						0		RST_to_CKE						
W																																	
Reset	0	0	0	0	0	0	0	0	1	0	0	1	1	1	1	1	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1	0

**MMDC\_MDOR field descriptions**

Field	Description
31–24 Reserved	This read-only field is reserved and always has the value 0.
23–16 tXPR	DDR3: CKE HIGH to a valid command.  This field is not relevant in LPDDR2 mode.  DDR3: As defined in timing parameter table.  0x0 Reserved 0x1 2 cycles

*Table continues on the next page...*

### MMDC\_MDOR field descriptions (continued)

Field	Description
	0x2 3 cycles 0xFE 255 cycles 0xFF 256 cycles
15–14 Reserved	This read-only field is reserved and always has the value 0.
13–8 SDE_to_RST	DDR3: Time from SDE enable until DDR reset# is high. In LPDDR2 mode this field is not relevant . <b>NOTE:</b> Each cycle in this field is 15.258 us.  0x0 Reserved 0x1 Reserved 0x2 Reserved 0x3 1 cycles 0x4 2 cycles 0x10 14 cycles (Jedec value for DDR3) - total of 200 us 0x3E 60 cycles 0x3F 61 cycles
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–0 RST_to_CKE	DDR3: Time from SDE enable to CKE rise. In case that DDR reset# is low, will wait until it's high and thenwait this period until rising CKE. (JEDEC value is 500 us) LPDDR2: Idle time ater first CKE assertion. (JEDEC value is 200 us) <b>NOTE:</b> Each cycle in this field is 15.258 us.  0x0 Reserved 0x1 Reserved 0x2 Reserved 0x3 1 cycles 0x10 14 cycles (JEDEC value for LPDDR2) - total of 200 us 0x23 33 cycles (JEDEC value for DDR3) - total of 500 us 0x3E 60 cycles 0x3F 61 cycles

### 32.12.12 MMDC Core MRR Data Register (MMDC\_MDMRR)

This register contains data that was collected after issuing MRR command. The data in this register is valid only when MDSCR[MRR\_READ\_DATA\_VALID] is set to "1".

This register is relevant only in LPDDR2 mode. For further information see [LPDDR2 Refresh Rate Update and Timing Derating](#) .



Address: 21B\_0000h base + 34h offset = 21B\_0034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MRR_READ_DATA3								MRR_READ_DATA2								MRR_READ_DATA1								MRR_READ_DATA0							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MDMRR field descriptions**

Field	Description
31–24 MRR_READ_DATA3	MRR DATA that arrived on DQ[31:24]
23–16 MRR_READ_DATA2	MRR DATA that arrived on DQ[23:16]
15–8 MRR_READ_DATA1	MRR DATA that arrived on DQ[15:8]
7–0 MRR_READ_DATA0	MRR DATA that arrived on DQ[7:0]

### 32.12.13 MMDC Core Timing Configuration Register 3 (MMDC\_MDCFG3LP)

This register is relevant only for LPDDR2 mode.

Address: 21B\_0000h base + 38h offset = 21B\_0038h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								RC_LP								0				tRCD_LP				tRPpb_LP				tRPab_LP			
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MDCFG3LP field descriptions**

Field	Description
31–22 Reserved	This read-only field is reserved and always has the value 0.
21–16 RC_LP	Active to Active or Refresh command period (same bank). (This field is valid only for LPDDR2 memories)  0x0    1 clock 0x1    2 clocks 0x2    3 clocks

*Table continues on the next page...*

### MMDC\_MDCFG3LP field descriptions (continued)

Field	Description
	0x3E 63 clocks 0x3F Reserved
15–12 Reserved	This read-only field is reserved and always has the value 0.
11–8 tRCD_LP	Active command to internal read or write delay time (same bank). (This field is valid only for LPDDR2 memories)  0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0xE 15 clocks 0xF Reserved
7–4 tRPpb_LP	Precharge (per bank) command period (same bank). (This field is valid only for LPDDR2 memories)  0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0xE 15 clocks 0xF Reserved
3–0 tRPab_LP	Precharge (all banks) command period. (This field is valid only for LPDDR2 memories)  0x0 1 clock 0x1 2 clocks 0x2 3 clocks 0xE 15 clocks 0xF Reserved

### 32.12.14 MMDC Core MR4 Derating Register (MMDC\_MDMR4)

This register is relevant only for LPDDR2 mode. It is used to dynamically change certain values depending on MR4 read result, which is based on memory temperature sensor result.

Address: 21B\_0000h base + 3Ch offset = 21B\_003Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	0								tRRD_DE	tRP_DE	tRAS_DE	tRC_DE	tRCD_DE	0		UPDATE_DE_ACK	UPDATE_DE_REQ
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### MMDC\_MDMR4 field descriptions

Field	Description
31–9 Reserved	This read-only field is reserved and always has the value 0.
8 tRRD_DE	tRRD derating value. 0 Original tRRD is used. 1 tRRD is derated in 1 cycle.
7 tRP_DE	tRP derating value. 0 Original tRP is used. 1 tRP is derated in 1 cycle.
6 tRAS_DE	tRAS derating value. 0 Original tRAS is used. 1 tRAS is derated in 1 cycle.
5 tRC_DE	tRC derating value. 0 Original tRC is used. 1 tRC is derated in 1 cycle.
4 tRCD_DE	tRCD derating value. 0 Original tRCD is used. 1 tRCD is derated in 1 cycle.
3–2 Reserved	This read-only field is reserved and always has the value 0.
1 UPDATE_DE_ACK	Update Derated Values Acknowledge. This read only bit will be cleared upon UPDATE_DE_REQ assertion and will be set after the new values are taken.
0 UPDATE_DE_REQ	Update Derated Values Request. This read modify write field is automatically cleared after the request is issued. 0 Do nothing. 1 Request to update the following values: tRRD, tRCD, tRP, tRC, tRAS and refresh related fields(MDREF register): REF_CNT, REF_SEL, REFR

### 32.12.15 MMDC Core Address Space Partition Register (MMDC\_MDASP)

This register defines the partitioning between chip select 0 and chip select 1. For further information see [Chip select settings](#).

Address: 21B\_0000h base + 40h offset = 21B\_0040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																CS0_END															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1

#### MMDC\_MDASP field descriptions

Field	Description
31–7 Reserved	This read-only field is reserved and always has the value 0.
6–0 CS0_END	CS0_END. Defines the absolute last address associated with CS0 with increments of 256Mb. CS0_END=AXI_ADDRESS[31:25] bits.  000_0000 256Mb 000_0001 512Mb 001_1111 8Gb (1GB) 011_1111 16Gb (2GB) - default 111_1111 32Gb (4GB)

## 32.12.16 MMDC Core AXI Reordering Control Register (MMDC\_MAARCR)

This register determines the values of the weights used for the re-ordering arbitration engine. For further information see [Performance](#).

Address: 21B\_0000h base + 400h offset = 21B\_0400h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ARCR_SEC_ERR_LOCK	ARCR_SEC_ERR_EN	Reserved	ARCR_EXC_ERR_EN	Reserved				ARCR_RCH_EN	Reserved	ARCR_PAG_HIT			Reserved	ARCR_ACC_HIT	
W																
Reset	0	1	0	1	0	0	0	1	0	1	0	0	0	0	1	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				ARCR_DYN_JMP				ARCR_DYN_MAX				ARCR_GUARD			
W																
Reset	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0

**MMDC\_MAARCR field descriptions**

Field	Description
31 ARCR_SEC_ERR_LOCK	Once set, this bit locks ARCR_SEC_ERR_EN and prevents from its updating. This bit can be only cleared by reset Default value is 0x0 - encoding 0 (unlocked) 0 ARCR_SEC_ERR_EN is unlocked, so can be updated any moment 1 ARCR_SEC_ERR_EN is locked, so it can't be updated
30 ARCR_SEC_ERR_EN	This bit defines whether security read/write access violation result in SLV Error response or in OKAY response Default value is 0x1 - encoding 1(response is SLV Error, rresp/bresp=2'b10) 0 security violation results in OKAY response (rresp/bresp=2'b00) 1 security violation results in SLAVE Error response (rresp/bresp=2'b10)

Table continues on the next page...

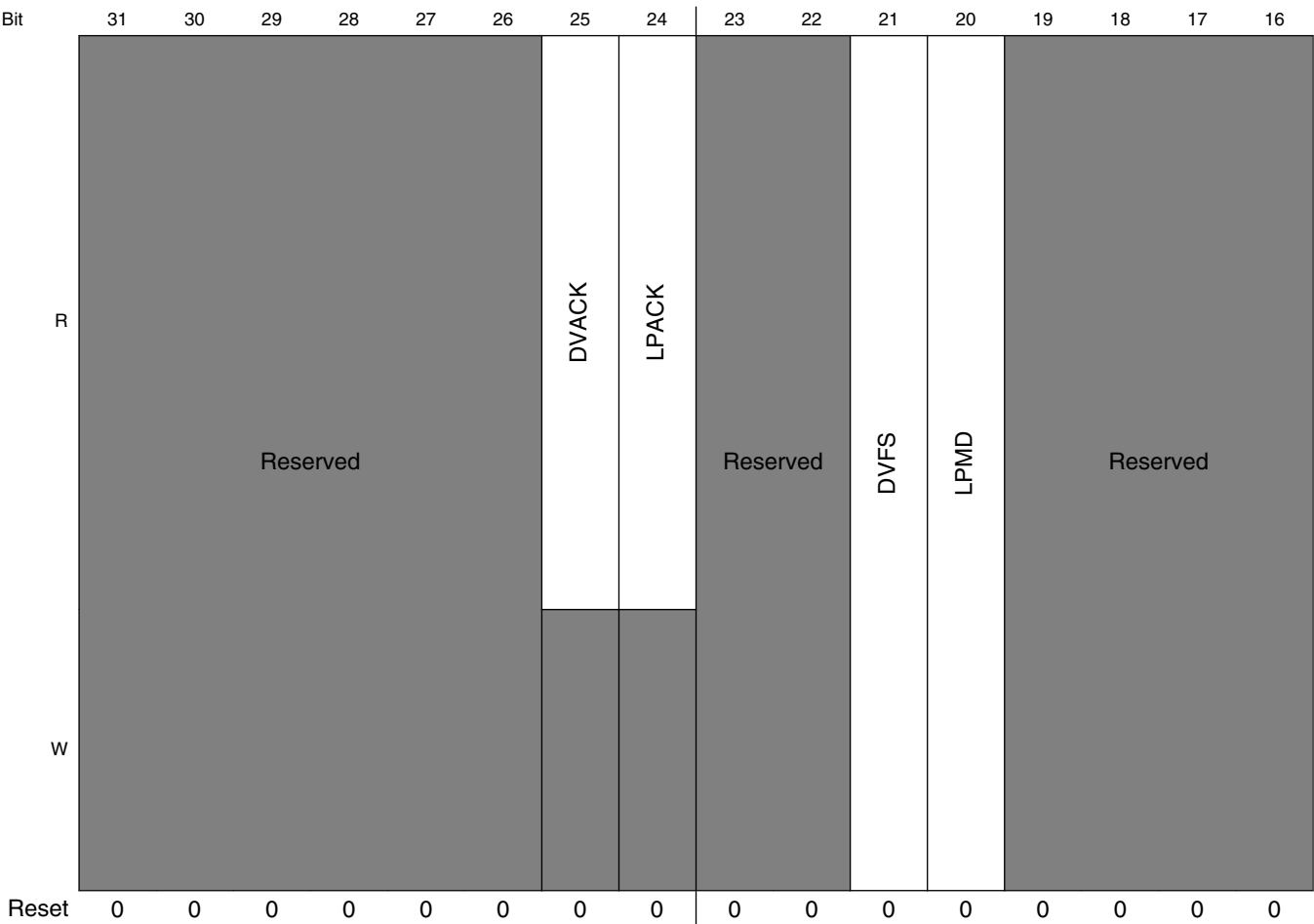
**MMDC\_MAARCR field descriptions (continued)**

Field	Description
29 -	This field is reserved. Reserved
28 ARCR_EXC_ERR_EN	This bit defines whether exclusive read/write access violation of AXI 6.2.4 rule result in SLV Error response or in OKAY response Default value is 0x1 - encoding 1(response is SLV Error)  0 violation of AXI exclusive rules (6.2.4) result in OKAY response (rresp/bresp=2'b00) 1 violation of AXI exclusive rules (6.2.4) result in SLAVE Error response (rresp/bresp=2'b10)
27–25 -	This field is reserved. Reserved
24 ARCR_RCH_EN	This bit defines whether Real time channel is activated and bypassed all other pending accesses, So accesses with QoS=='F' will be granted the highest priority in the optimization/reordering mechanism Default value is 0x1 - encoding 1 (Enabled)  0 normal prioritization, no bypassing 1 accesses with QoS=='F' bypass the arbitration
23 -	This field is reserved. Reserved
22–20 ARCR_PAG_HIT	ARCR Page Hit Rate. This value will be added by the optimization/reordering mechanism to any pending access that is targeted to an open DDR row. Default value of ARCR_PAG_HIT is 0x00100 - encoding 4.
19 -	This field is reserved. Reserved
18–16 ARCR_ACC_HIT	ARCR Access Hit Rate. This value will be added by the optimization/reordering mechanism to any pending access that has the same access type (read/write) as the previous access. Default value of is ARCR_ACC_HIT 0x0010 - encoding 2.
15–12 -	This field is reserved. Reserved
11–8 ARCR_DYN_JMP	ARCR Dynamic Jump. Each time an access wasn't chosen by the optimization/reordering mechanism then its dynamic score will be incremented by ARCR_DYN_JMP value. <b>NOTE:</b> Setting ARCR_DYN_JMP may cause starvation of low priority accesses <b>NOTE:</b> ARCR_DYN_JMP must be smaller than ARCR_DYN_MAX Default ARCR_DYN_JMP value is 0x0001 - encoding 1
7–4 ARCR_DYN_MAX	ARCR Dynamic Maximum. ARCR_DYN_MAX is the maximum dynamic score value that each access inside the optimization/reordering mechanism can get.  0000 0 0001 1 1111 15 (default)
3–0 ARCR_GUARD	ARCR Guard. After an access reached the maximum dynamic score value, it will wait additional ARCR_GUARD arbitration cycles and then will gain the highest priority in the optimization/reordering mechanism.  0000 15 (default) 0001 16 1111 30

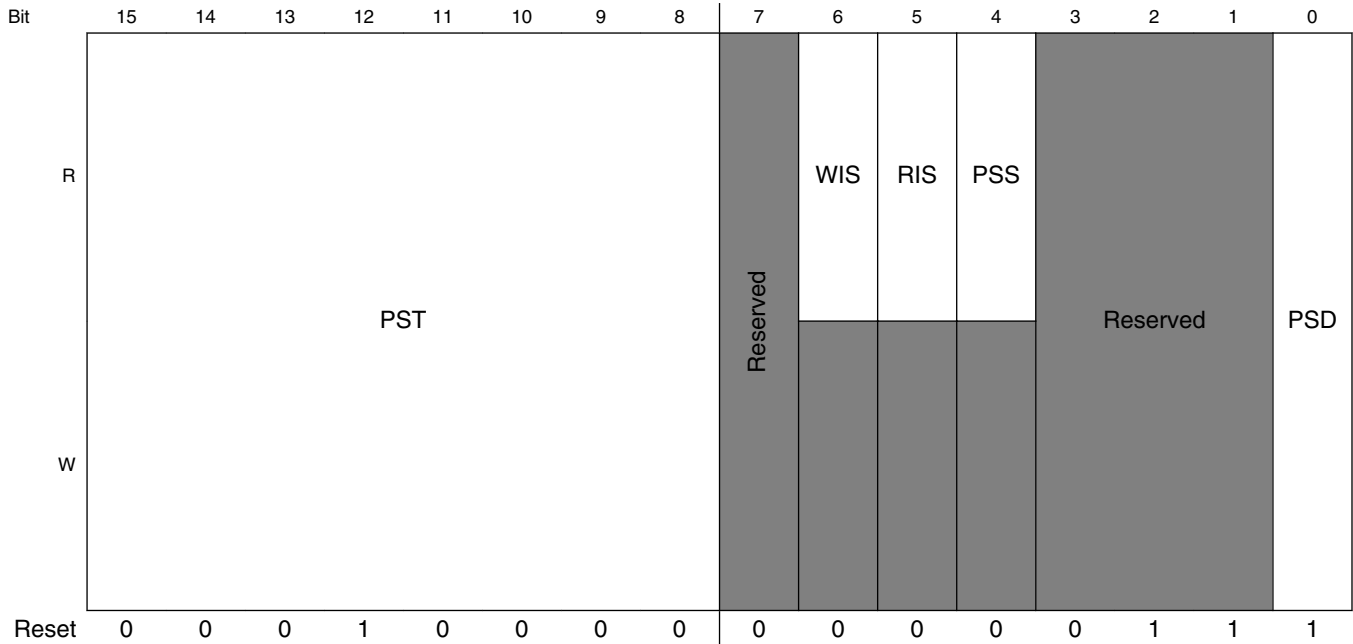
### 32.12.17 MMDC Core Power Saving Control and Status Register (MMDC\_MAPSR)

The MAPSR determines the power saving features of MMDC. For further information see [Power Saving and Clock Frequency Change modes](#) .

Address: 21B\_0000h base + 404h offset = 21B\_0404h





**MMDC\_MAPSR field descriptions**

Field	Description
31–26 -	This field is reserved. Reserved
25 DVACK	General DVFS acknowledge. This read only bit indicates whether a dvfs acknowledge was asserted and that MMDC is in self-refresh mode
24 LPACK	General low-power acknowledge. This read only bit indicates whether a low-power acknowledge was asserted and that MMDC is in self-refresh mode
23–22 -	This field is reserved. Reserved
21 DVFS	General DVFS request. SW request for DVFS. Assertion of this bit will yield in self-refresh entry sequence 0 no dvfs request 1 dvfs request
20 LPMD	General LPMD request. SW request for LPMD. Assertion of this bit will yield in self-refresh entry sequence 0 no lpmd request 1 lpmd request
19–16 -	This field is reserved. Reserved
15–8 PST	Automatic Power saving timer. Valid only when PSD is set to "0". When the MMDC is idle for amount of cycles specified in that field then the DDR device will be entered automatically into self-refresh mode. The real value which is used is register-value multiplied by 64.  00000000 Reserved - this value is forbidden. 00000001 timer is configured to 64 clock cycles. 00000010 timer is configured to 128 clock cycles.

*Table continues on the next page...*

## MMDC\_MAPSR field descriptions (continued)

Field	Description
	00010000 (Default)- 1024 clock cycles. 11111111 timer clock is configured to 16320 clock cycles.
7 -	This field is reserved. Reserved.
6 WIS	Write Idle Status. This read only bit indicates whether write request buffer is idle (empty) or not. 0 idle 1 not idle
5 RIS	Read Idle Status. This read only bit indicates whether read request buffer is idle (empty) or not. 0 idle 1 not idle
4 PSS	Power Saving Status. This read only bit indicates whether the MMDC is in automatic power saving mode. 0 not in power saving 1 power saving
3-1 -	This field is reserved. Reserved.
0 PSD	Automatic Power Saving Disable. When the value of PSD is "0" (i.e automatic power saving is enabled) then the PST is activated and MMDC will enter automatically to self-refresh while the number of idle cycle reached.  <b>NOTE:</b> This bit must be disabled (i.e set to "1") during calibration process  0 power saving enabled 1 power saving disabled (default)

### 32.12.18 MMDC Core Exclusive ID Monitor Register0 (MMDC\_MAEXIDR0)

This register defines the ID to be monitored for exclusive accesses of monitor0 and monitor1. For further information see [Exclusive accesses handling](#).

Address: 21B\_0000h base + 408h offset = 21B\_0408h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EXC_ID_MONITOR1																EXC_ID_MONITOR0															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**MMDC\_MAEXIDR0 field descriptions**

Field	Description
31–16 EXC_ID_MONITOR1	This feild defines ID for Exclusive monitor#1. Default value is 0x0020
15–0 EXC_ID_MONITOR0	This feild defines ID for Exclusive monitor#0. Default value is 0x0000

**32.12.19 MMDC Core Exclusive ID Monitor Register1 (MMDC\_MAEXIDR1)**

This register defines the ID to be monitored for exclusive accesses of monitor2 and monitor3. For further information see [Exclusive accesses handling](#) .

Address: 21B\_0000h base + 40Ch offset = 21B\_040Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EXC_ID_MONITOR3																EXC_ID_MONITOR2															
W	EXC_ID_MONITOR3																EXC_ID_MONITOR2															
Reset	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0

**MMDC\_MAEXIDR1 field descriptions**

Field	Description
31–16 EXC_ID_MONITOR3	This feild defines ID for Exclusive monitor#3. Default value is 0x0060
15–0 EXC_ID_MONITOR2	This feild defines ID for Exclusive monitor#2. Default value is 0x0040

## 32.12.20 MMDC Core Debug and Profiling Control Register 0 (MMDC\_MADPCR0)

Address: 21B\_0000h base + 410h offset = 21B\_0410h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								0				CYC_OVF	PRF_FRZ	DBG_RST	DBG_EN
W							SBS	SBS_EN					w1c			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MADPCR0 field descriptions**

Field	Description
31–10 Reserved	This read-only field is reserved and always has the value 0.
9 SBS	Step By Step trigger. If SBS_EN is set to "1" then dispatching AXI pending access toward the DDR will done only if this bit is set to "1", otherwise no access will be dispatched toward the DDR. This bit is cleared when the pending access has been issued toward the DDR device.  1 Lanuch AXI pending access toward the DDR 0 No access will be launched toward the DDR
8 SBS_EN	Step By Step debug Enable. Enable step by step mode. Every time this mechanism is enabled then setting SBS to "1" will dispatch one pending AXI access to the DDR and in parallel its attributes will be observed in the status registes (MASBS0 and MASBS1). For further information see <a href="#">Step By Step (SBS) software monitor</a> .  0 disable 1 enable
7–4 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**MMDC\_MADPCR0 field descriptions (continued)**

Field	Description
3 CYC_OVF	Total Profiling Cycles Count Overflow. When profiling mechanism is enabled (DBG_EN is set to "1") then this bit is asserted when overflow of CYC_COUNT occurred. Cleared by writing 1 to it.  0 no overflow 1 overflow
2 PRF_FRZ	Profiling freeze. When this bit is asserted then the profiling mechanism will be freezed and the associated status registers ( MADPSR0-MADPSR5) will hold the the current profiling values.  0 profiling counters are not frozen 1 profiling counters are frozen
1 DBG_RST	Debug and Profiling Reset. Reset all debug and profiling counters and components.  0 no reset 1 reset
0 DBG_EN	Debug and Profiling Enable. Enable debug and profiling mechanism. When this bit is asserted then the MMDC will perform a profiling based on the ID that is configured to MADPCR1. Upon assertion of PRF_FRZ the profiling will be freezed and the profiling results will be sampled to the status registers (MADPSR0-MADPSR5). For further information see <a href="#">MMDC Profiling</a> . default is "disable"  0 disable 1 enable

**32.12.21 MMDC Core Debug and Profiling Control Register 1 (MMDC\_MADPCR1)**

Address: 21B\_0000h base + 414h offset = 21B\_0414h

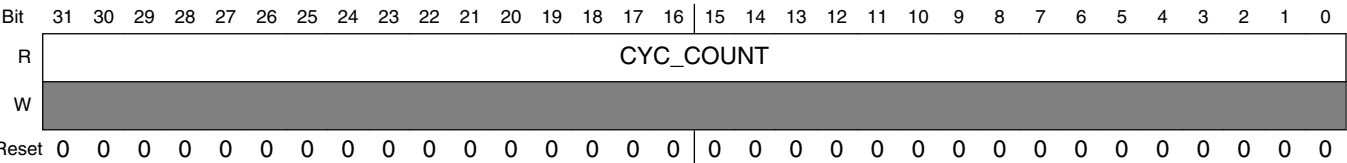
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PRF_AXI_ID_MASK																PRF_AXI_ID															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**MMDC\_MADPCR1 field descriptions**

Field	Description
31–16 PRF_AXI_ID_MASK	Profiling AXI ID Mask. AXI ID bits which masked by this value are chosen for profiling.  1 AXI ID specific bit is chosen for profiling 0 AXI ID specific bit is ignored (don't care)
15–0 PRF_AXI_ID	Profiling AXI ID. AXI IDs that matches a bit-wise AND logic operation between PRF_AXI_ID and PRF_AXI_ID_MASK are chosen for profiling.  Default value is 0x0, to choose any ID-s for profiling

32.12.22 MMDC Core Debug and Profiling Status Register 0 (MMDC\_MADPSR0)

Address: 21B\_0000h base + 418h offset = 21B\_0418h



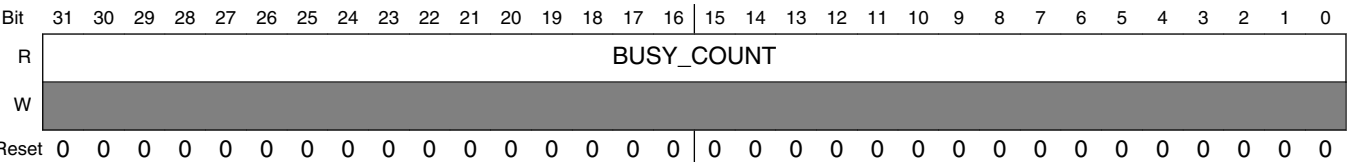
MMDC\_MADPSR0 field descriptions

Field	Description
31–0 CYC_COUNT	Total Profiling cycle Count. This field reflects the total cycle count in case the profiling mechanism is enabled from assertion of DBG_EN and until PRF_FRZ is asserted

32.12.23 MMDC Core Debug and Profiling Status Register 1 (MMDC\_MADPSR1)

The register reflects the total cycles during which the MMDC state machines were busy (both writes and reads). This information can be used for DDR Utilization calculation.

Address: 21B\_0000h base + 41Ch offset = 21B\_041Ch



MMDC\_MADPSR1 field descriptions

Field	Description
31–0 BUSY_COUNT	Profiling Busy Cycles Count. This field reflects the total number of cycles where the MMDC read and write state machines were busy during the profiling period. Can be used for DDR utilization calculations

### 32.12.24 MMDC Core Debug and Profiling Status Register 2 (MMDC\_MADPSR2)

This register reflects the total number of read accesses (per AXI ID) toward MMDC.

Address: 21B\_0000h base + 420h offset = 21B\_0420h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RD_ACC_COUNT																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### MMDC\_MADPSR2 field descriptions

Field	Description
31–0 RD_ACC_COUNT	Profiling Read Access Count. This register reflects the total number of read accesses (per AXI ID) toward MMDC.

### 32.12.25 MMDC Core Debug and Profiling Status Register 3 (MMDC\_MADPSR3)

This register reflects the total number of write accesses (per AXI ID) toward MMDC.

Address: 21B\_0000h base + 424h offset = 21B\_0424h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WR_ACC_COUNT																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### MMDC\_MADPSR3 field descriptions

Field	Description
31–0 WR_ACC_COUNT	Profiling Write Access Count. This register reflects the total number of write accesses (per AXI ID) toward MMDC.

### 32.12.26 MMDC Core Debug and Profiling Status Register 4 (MMDC\_MADPSR4)

This register reflects the total number of bytes that were transferred during read access (per AXI ID) toward MMDC.

Address: 21B\_0000h base + 428h offset = 21B\_0428h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RD_BYTES_COUNT																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### MMDC\_MADPSR4 field descriptions

Field	Description
31–0 RD_BYTES_COUNT	Profiling Read Bytes Count. This register reflects the total number of bytes that were transferred during read access (per AXI ID) toward MMDC.

### 32.12.27 MMDC Core Debug and Profiling Status Register 5 (MMDC\_MADPSR5)

This register reflects the total number of bytes that were transferred during write access (per AXI ID) toward MMDC.

Address: 21B\_0000h base + 42Ch offset = 21B\_042Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WR_BYTES_COUNT																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### MMDC\_MADPSR5 field descriptions

Field	Description
31–0 WR_BYTES_COUNT	Profiling Write Bytes Count. This register reflects the total number of bytes that were transferred during write access (per AXI ID) toward MMDC.



## 32.12.28 MMDC Core Step By Step Address Register (MMDC\_MASBS0)

Address: 21B\_0000h base + 430h offset = 21B\_0430h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBS_ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MASBS0 field descriptions**

Field	Description
31–0 SBS_ADDR	Step By Step Address. These bits reflect the address of the pending request in case of step by step mode.

## 32.12.29 MMDC Core Step By Step Address Attributes Register (MMDC\_MASBS1)

Address: 21B\_0000h base + 434h offset = 21B\_0434h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SBS_AXI_ID															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SBS_LEN			SBS_BUFF	SBS_BURST		SBS_SIZE			SBS_PROT			SBS_LOCK		SBS_TYPE	SBS_VLD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MASBS1 field descriptions**

Field	Description
31–16 SBS_AXI_ID	Step By Step AXI ID. These bits reflect the AXI ID of the pending request in case of step by step mode.
15–13 SBS_LEN	Step By Step Length. These bits reflect the AXI LENGTH of the pending request in case of step by step mode.  000 burst of length 1 001 burst of length 2 111 burst of length 8
12 SBS_BUFF	Step By Step Buffered. This bit reflect the AXI CACHE[0] of the pending request in case of step by step mode. Relevant only for write requests

*Table continues on the next page...*

### MMDC\_MASBS1 field descriptions (continued)

Field	Description
11–10 SBS_BURST	Step By Step Burst. These bits reflect the AXI BURST of the pending request in case of step by step mode.  00 FIXED 01 INCR burst 10 WRAP burst 11 reserved
9–7 SBS_SIZE	Step By Step Size. These bits reflect the AXI SIZE of the pending request in case of step by step mode.  000 8 bits 001 16 bits 010 32 bits 011 64 bits 100 128bits 101-111 Reserved
6–4 SBS_PROT	Step By Step Protection. These bits reflect the AXI PROT of the pending request in case of step by step mode.
3–2 SBS_LOCK	Step By Step Lock. These bits reflect the AXI LOCK of the pending request in case of step by step mode.
1 SBS_TYPE	Step By Step Request Type. These bits reflect the type (read/write) of the pending request in case of step by step mode.  0 write 1 read
0 SBS_VLD	Step By Step Valid. This bit reflects whether there is a pending request in case of step by step mode.  0 not valid 1 valid

## 32.12.30 MMDC Core General Purpose Register (MMDC\_MAGENP)

This register is a general 32 bit read/write register.

Address: 21B\_0000h base + 440h offset = 21B\_0440h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### MMDC\_MAGENP field descriptions

Field	Description
31–0 GP31_GP0	General purpose read/write bits.

### 32.12.31 MMDC PHY ZQ HW control register (MMDC\_MPZQHWCTRL)

Address: 21B\_0000h base + 800h offset = 21B\_0800h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	ZQ_EARLY_COMPARATOR_EN_TIMER					0	TZQ_CS			TZQ_OPER			TZQ_INIT			ZQ_HW_FOR
W																
Reset	1	0	1	0	0	0	0	1	0	0	1	1	1	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ZQ_HW_PD_RES					ZQ_HW_PU_RES					ZQ_HW_PER					ZQ_MODE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### MMDC\_MPZQHWCTRL field descriptions

Field	Description
31–27 ZQ_EARLY_COMPARATOR_EN_TIMER	ZQ early comparator enable timer. This timer defines the interval between the warming up of the comparator of the i.MX ZQ calibration pad and the beginning of the ZQ calibration process with the pad 0x0 - 0x6 Reserved 0x7 8 cycles 0x14 21 cycles (Default) 0x1E 31 cycles 0x1F 32 cycles
26 Reserved	This read-only field is reserved and always has the value 0.
25–23 TZQ_CS	Device ZQ short time. This field holds the number of cycles that are required by the external DDR device to perform ZQ short calibration. Upon driving the command to the DDR device then no further accesses will be issued to the DDR device till satisfying that time. <b>NOTE:</b> In LPDDR2 the ZQ short time is taken from MPZQLP2CTL[ZQ_LP2_HW_ZQCS] <b>NOTE:</b> This field should not be update during ZQ calibration. 000 Reserved 001 Reserved 010 128 cycles (Default) 011 256 cycles 100 512 cycles 101 1024 cycles 110- 111 Resreved
22–20 TZQ_OPER	Device ZQ long/oper time. This field holds the number of cycles that are required by the external DDR device to perform ZQ long calibration except the first ZQ long command that is issued after reset. Upon driving the command to the DDR device then no further accesses will be issued to the DDR device till satisfying that time. <b>NOTE:</b> In LPDDR2 the ZQ oper time is taken from MPZQLP2CTL[ZQ_LP2_HW_ZQCL]

Table continues on the next page...

### MMDC\_MPZQHWCTRL field descriptions (continued)

Field	Description
	<p><b>NOTE:</b> This field should not be update during ZQ calibration.</p> <p>000 Reserved</p> <p>001 Reserved</p> <p>010 128 cycles</p> <p>011 256 cycles - Default (JEDEC value for DDR3)</p> <p>100 512 cycles</p> <p>101 1024 cycles</p> <p>110- 111 Resreved</p>
19–17 TZQ_INIT	<p>Device ZQ long/init time. This field holds the number of cycles that are required by the external DDR device to perform ZQ long calibration right after reset. Upon driving the command to the DDR device then no further accesses will be issued to the DDR device till satisfying that time.</p> <p><b>NOTE:</b> In LPDDR2 the ZQ init time is taken from MPZQLP2CTL[ZQ_LP2_HW_ZQINIT]</p> <p><b>NOTE:</b> This field should not be update during ZQ calibration.</p> <p>000 Reserved</p> <p>001 Reserved</p> <p>010 128 cycles</p> <p>011 256 cycles</p> <p>100 512 cycles - Default (JEDEC value for DDR3)</p> <p>101 1024 cycles</p> <p>110- 111 Resreved</p>
16 ZQ_HW_FOR	<p>Force ZQ automatic calibration process with the i.MX ZQ calibration pad. When this bit is asserted then the MMDC will issue one ZQ automatic calibration process with the i.MX ZQ calibration pad. It is the user responsibility to make sure that all the accesses to DDR will be finished before asserting this bit using CON_REQ/CON_ACK mechanism. HW will negate this bit upon completion of the ZQ calibration process. Upon negation of this bit the ZQ HW calibration pull-up and pull-down results (ZQ_HW_PU_RES and ZQ_HW_PD_RES respectively) are valid</p> <p><b>NOTE:</b> In order to enable this bit ZQ_MODE must be set to either "1" or "3"</p>
15–11 ZQ_HW_PD_RES	<p>ZQ HW calibration pull-down result. This field holds the pull-down resistor value calculated at the end of the ZQ automatic calibration process with the i.MX ZQ calibration pad.</p> <p>00000 Max. resistance.</p> <p>11111 Min. resistance.</p>
10–6 ZQ_HW_PU_RES	<p>ZQ automatic calibration pull-up result. This field holds the pull-up resistor value calculated at the end of the ZQ automatic calibration process with the i.MX ZQ calibration pad.</p> <p>00000 Min. resistance.</p> <p>11111 Max. resistance.</p>
5–2 ZQ_HW_PER	<p>ZQ periodic calibration time. This field determines how often the periodic ZQ calibration is performed.</p> <p>This field is applied for both ZQ short calibration and ZQ automatic calibration process with i.MX ZQ calibration pad. Whenever this timer is expired then according to ZQ_MODE the ZQ automatic calibration process with the i.MX ZQ calibration pad will be issued and/or short/long command will be issued to the external DDR device.</p> <p>This field is ignored if ZQ_MODE equals "00"</p> <p>0000 ZQ calibration is performed every 1 ms.</p> <p>0001 ZQ calibration is performed every 2 ms.</p>

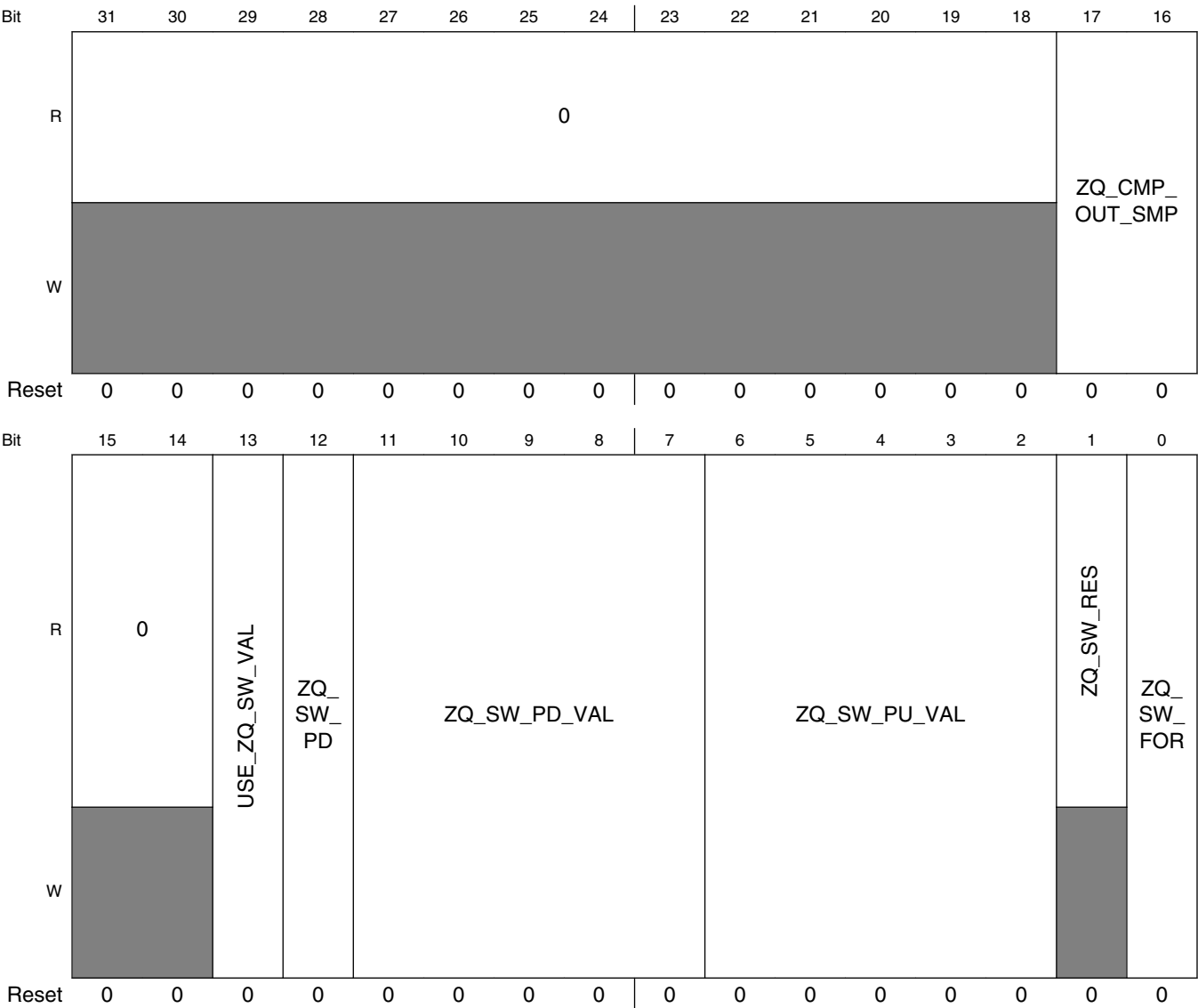
*Table continues on the next page...*

**MMDC\_MPZQHWCTRL field descriptions (continued)**

Field	Description
	0010 ZQ calibration is performed every 4 ms. 1010 ZQ calibration is performed every 1 ms. 1110 ZQ calibration is performed every 16 ms. 1111 ZQ calibration is performed every 32 ms.
1–0 ZQ_MODE	ZQ calibration mode: 0x0 No ZQ calibration is issued. (Default) 0x1 ZQ calibration is issued to i.MX ZQ calibration pad together with ZQ long command to the external DDR device only when exiting self refresh. 0x2 ZQ calibration command long/short is issued only to the external DDR device periodically and when exiting self refresh 0x3 ZQ calibration is issued to i.MX ZQ calibration pad together with ZQ calibration command long/short to the external DDR device periodically and when exiting self refresh

### 32.12.32 MMDC PHY ZQ SW control register (MMDC\_MPZQSWCTRL)

Address: 21B\_0000h base + 804h offset = 21B\_0804h



MMDC\_MPZQSWCTRL field descriptions

Field	Description
31–18 Reserved	This read-only field is reserved and always has the value 0.
17–16 ZQ_CMP_OUT_SMP	Defines the amount of cycles between driving the ZQ signals to the ZQ pad and till sampling the comparator enable output while performing ZQ calibration process with the i.MX ZQ calibration pad  00 7 cycles 01 15 cycles

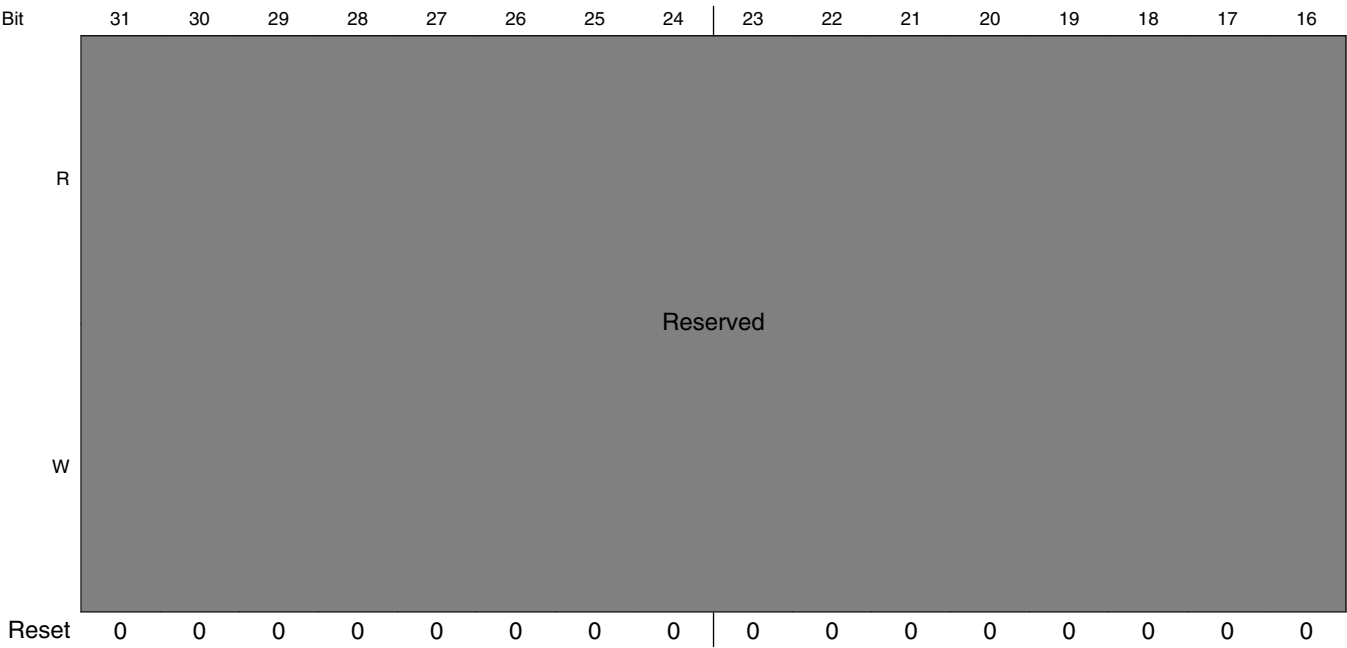
Table continues on the next page...

**MMDC\_MPZQSWCTRL field descriptions (continued)**

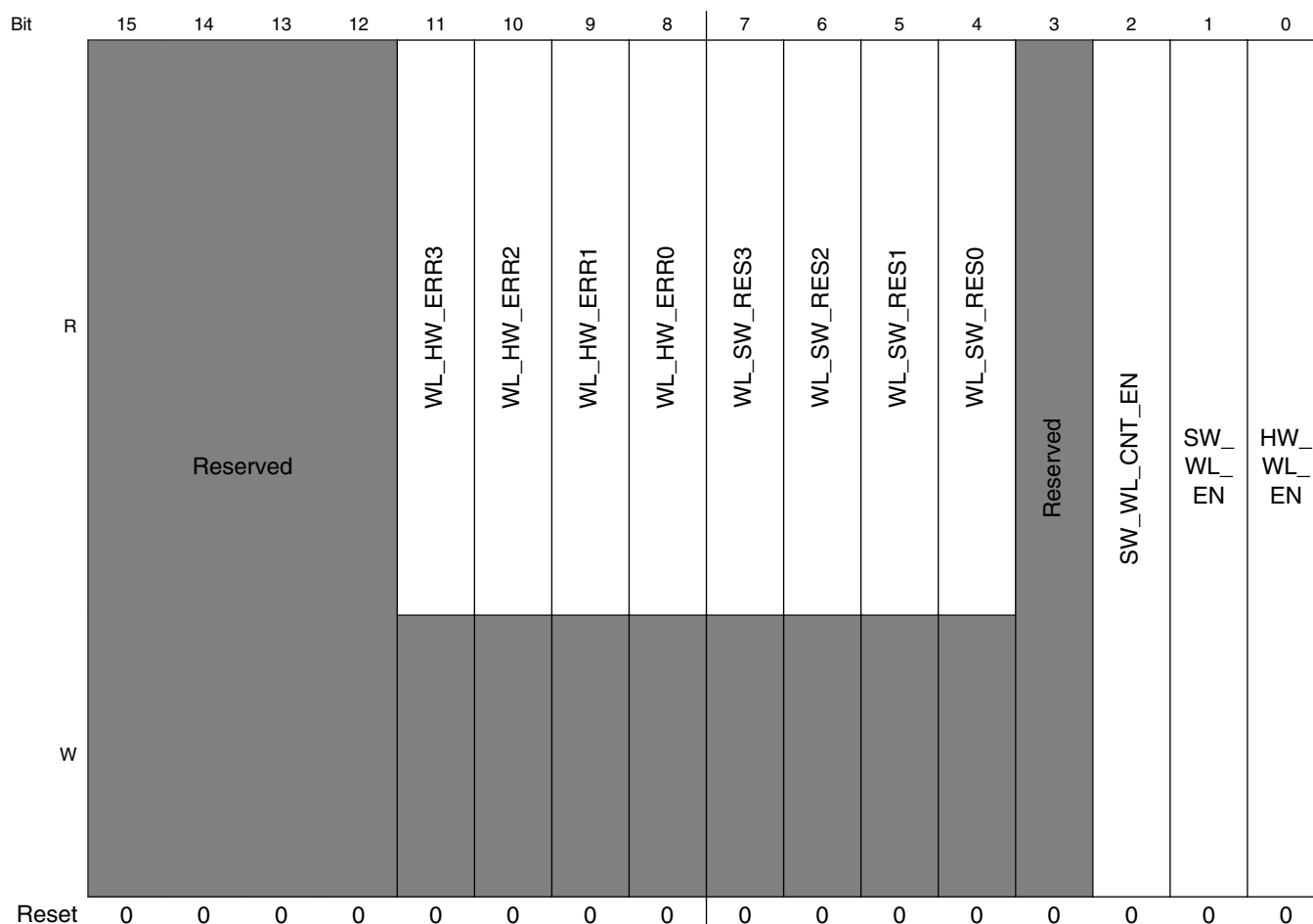
Field	Description
	10 23 cycles 11 31 cycles
15–14 Reserved	This read-only field is reserved and always has the value 0.
13 USE_ZQ_SW_VAL	Use SW ZQ configured value for I/O pads resistor controls. This bit selects whether ZQ SW value or ZQ HW value will be driven to the I/O pads resistor controls. By default this bit is cleared and MMDC drives the HW ZQ status bits on the resistor controls of the I/O pads.  <b>NOTE:</b> This bit should not be updated during ZQ calibration.  0 Fields ZQ_HW_PD_VAL & ZQ_HW_PU_VAL will be driven to I/O pads resistor controls. 1 Fields ZQ_SW_PD_VAL & ZQ_SW_PU_VAL will be driven to I/O pads resistor controls.
12 ZQ_SW_PD	ZQ software PU/PD calibration. This bit determines the calibration stage (PU or PD).  0 PU resistor calibration 1 PD resistor calibration
11–7 ZQ_SW_PD_VAL	ZQ software pull-down resistance. This field determines the value of the PD resistor during SW ZQ calibration.  00000 Max. resistance. 11111 Min. resistance.
6–2 ZQ_SW_PU_VAL	ZQ software pull-up resistance. This field determines the value of the PU resistor during SW ZQ calibration.  00000 Min. resistance. 11111 Max. resistance.
1 ZQ_SW_RES	ZQ software calibration result. This bit reflects the ZQ calibration voltage comparator value.  0 Current ZQ calibration voltage is less than VDD/2. 1 Current ZQ calibration voltage is more than VDD/2
0 ZQ_SW_FOR	ZQ SW calibration enable. This bit when asserted enables ZQ SW calibration. HW negates this bit upon completion of the ZQ SW calibration. Upon negation of this bit the ZQ SW calibration result (i.e ZQ_SW_RES) is valid

### 32.12.33 MMDC PHY Write Leveling Configuration and Error Status Register (MMDC\_MPWLGCR)

Address: 21B\_0000h base + 808h offset = 21B\_0808h







MMDC\_MPWLGCR field descriptions

Field	Description
31–12 -	This field is reserved. Reserved
11 WL_HW_ERR3	Byte3 write-leveling HW calibration error. This bit is asserted when an error was found on byte3 during write-leveling HW calibration.  This bit is valid only upon completion of the write-leveling HW calibration (i.e HW_WL_EN bit is de-asserted)  0 No error was found on byte3 during write-leveling HW calibration. 1 An error was found on byte3 during write-leveling HW calibration.
10 WL_HW_ERR2	Byte2 write-leveling HW calibration error. This bit is asserted when an error was found on byte2 during write-leveling HW calibration.  This bit is valid only upon completion of the write-leveling HW calibration (i.e HW_WL_EN bit is de-asserted)  0 No error was found on byte2 during write-leveling HW calibration. 1 An error was found on byte2 during write-leveling HW calibration.
9 WL_HW_ERR1	Byte1 write-leveling HW calibration error. This bit is asserted when an error was found on byte1 during write-leveling HW calibration.

Table continues on the next page...

### MMDC\_MPWLGCRC field descriptions (continued)

Field	Description
	<p>This bit is valid only upon completion of the write-leveling HW calibration (i.e HW_WL_EN bit is de-asserted)</p> <p>0 No error was found on byte1 during write-leveling HW calibration.</p> <p>1 An error was found on byte1 during write-leveling HW calibration.</p>
8 WL_HW_ERR0	<p>Byte0 write-leveling HW calibration error. This bit is asserted when an error was found on byte0 during write-leveling HW calibration.</p> <p>This bit is valid only upon completion of the write-leveling HW calibration (i.e HW_WL_EN bit is de-asserted)</p> <p>0 No error was found on byte0 during write-leveling HW calibration.</p> <p>1 An error was found on byte0 during write-leveling HW calibration.</p>
7 WL_SW_RES3	<p>Byte3 write-leveling software result. This bit reflects the value that is driven by the DDR device on DQ24 during SW write-leveling.</p> <p>0 DQS3 sampled low CK during SW write-leveling.</p> <p>1 DQS3 sampled high CK during SW write-leveling.</p>
6 WL_SW_RES2	<p>Byte2 write-leveling software result. This bit reflects the value that is driven by the DDR device on DQ16 during SW write-leveling.</p> <p>0 DQS2 sampled low CK during SW write-leveling.</p> <p>1 DQS2 sampled high CK during SW write-leveling.</p>
5 WL_SW_RES1	<p>Byte1 write-leveling software result. This bit reflects the value that is driven by the DDR device on DQ8 during SW write-leveling.</p> <p>0 DQS1 sampled low CK during SW write-leveling.</p> <p>1 DQS1 sampled high CK during SW write-leveling.</p>
4 WL_SW_RES0	<p>Byte0 write-leveling software result. This bit reflects the value that is driven by the DDR device on DQ0 during SW write-leveling.</p> <p>0 DQS0 sampled low CK during SW write-leveling.</p> <p>1 DQS0 sampled high CK during SW write-leveling.</p>
3 -	<p>This field is reserved.</p> <p>Reserved</p>
2 SW_WL_CNT_EN	<p>SW write-leveling count down enable. This bit when asserted set a certain delay of (25+15) cycles from the setting of SW_WL_EN and before driving the DQS to the DDR device. This bit should be asserted before the first SW write-leveling request and after issuing the write leveling MRS command</p> <p>0 MMDC doesn't count 25+15 cycles before issuing write-leveling DQS.</p> <p>1 MMDC counts 25+15 cycles before issuing write-leveling DQS.</p>
1 SW_WL_EN	<p>Write-Leveling SW enable. If this bit is asserted then the MMDC will perform one write-leveling iteration with the DDR device (assuming that Write-Leveling procedure is already enabled in the DDR device through MRS command). HW negate this bit upon completion of the SW write-leveling. Negation of this bit also points that the write-leveling SW calibration result is valid</p> <p><b>NOTE:</b> If this bit and the SW_WL_CNT_EN are enabled the MMDC counts 25 + 15 cycles before issuing the SW write-leveling DQS.</p>
0 HW_WL_EN	<p>Write-Leveling HW (automatic) enable. If this bit is asserted then the MMDC will perform the whole Write-Leveling sequence with the DDR device (assuming that Write-Leveling procedure is already enabled in the DDR device through MRS command). HW negates this bit upon completion of the HW write-leveling. Negation of this bit also points that the write-leveling HW calibration results are valid</p>

*Table continues on the next page...*

**MMDC\_MPWLGCR field descriptions (continued)**

Field	Description
	<b>NOTE:</b> Before issuing the first DQS the MMDC counts 25 + 15 cycles automatically as required by the standard.

### 32.12.34 MMDC PHY Write Leveling Delay Control Register 0 (MMDC\_MPWLDECTRL0)

Address: 21B\_0000h base + 80Ch offset = 21B\_080Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					WL_CYC_DEL1		WL_HC_DEL1	0	WL_DL_ABS_OFFSET1						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					WL_CYC_DELO		WL_HC_DELO	0	WL_DL_ABS_OFFSET0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MPWLDECTRL0 field descriptions**

Field	Description
31–27 Reserved	This read-only field is reserved and always has the value 0.
26–25 WL_CYC_DEL1	<p>Write leveling cycle delay for Byte 1. This field indicates whether a delay of 1 or 2 cycles between CK and write DQS is added to the delay that is indicated in the associated WR_DL_ABS_OFFSET and WL_HC_DEL. So the total delay is the sum of (WL_DL_ABS_OFFSET/256*cycle) + (WL_HC_DEL*half cycle) + (WL_CYC_DEL*cycle).</p> <p>When both SW write-leveling is enabled (i.e SW_WL_EN = 1) or HW write-leveling is enabled (i.e HW_WL_EN = 1 ) then this value will be taken as is and will be added to the associated delay that is configured in WL_DL_OFFSET and WL_HC_DEL.</p> <p>Note that in HW write-leveling this field is not used for indication, as in WL_DL_OFFSET and WL_HC_DEL, but for configuration.</p> <p>0 No delay is added.  1 1 cycle delay is added.  2 2 cycles delay is added.  3 Reserved.</p>
24 WL_HC_DEL1	Write leveling half cycle delay for Byte 1. This field indicates whether a delay of half cycle between CK and write DQS is added to the delay that is indicated in the associated WR_DL_ABS_OFFSET and

*Table continues on the next page...*

## MMDC\_MPWLDECTRL0 field descriptions (continued)

Field	Description
	<p>WL_CYC_DEL. So the total delay is the sum of <math>(WL\_DL\_ABS\_OFFSET/256 \times \text{cycle}) + (WL\_HC\_DEL \times \text{half cycle}) + (WL\_CYC\_DEL \times \text{cycle})</math>.</p> <p>When SW write-leveling is enabled (i.e SW_WL_EN = 1) then this value will be taken as is and will be added to the associated delay that is configured in WL_DL_OFFSET and WL_CYC_DEL. When HW write-leveling is enabled (i.e HW_WL_EN = 1 ) then this value will indicate (status) whether a delay of half cycle was added or not to the associated WL_DL_OFFSET and WL_CYC_DEL.</p> <p>0 No delay is added. 1 Half cycle delay is added.</p>
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 WL_DL_ABS_OFFSET1	<p>Absolute write-leveling delay offset for Byte 1. This field indicates the absolute delay between CK and write DQS of Byte1 with fractions of a clock period and up to half cycle. This value is process and frequency independent. The value of the delay can be calculated using the following equation <math>(WR\_DL\_ABS\_OFFSET1 / 256) \times \text{clock period}</math></p> <p>When SW write-leveling is enabled (i.e SW_WL_EN = 1) then this value will be taken as is to the associated delay-line. When HW write-leveling is enabled (i.e HW_WL_EN = 1 ) then this value will indicate (status) the value that is taken to the associated delay-line at the end of the write-leveling calibration.</p> <p><b>NOTE:</b> The delay-line has a resolution that may vary between device to device, therefore in some cases an increment of the delay by 1 step may be smaller than the delay-line resolution.</p>
15–11 Reserved	This read-only field is reserved and always has the value 0.
10–9 WL_CYC_DELO	<p>Write leveling cycle delay for Byte 0. This field indicates whether a delay of 1 or 2 cycles between CK and write DQS is added to the delay that is indicated in the associated WR_DL_ABS_OFFSET and WL_HC_DEL. So the total delay is the sum of <math>(WL\_DL\_ABS\_OFFSET/256 \times \text{cycle}) + (WL\_HC\_DEL \times \text{half cycle}) + (WL\_CYC\_DEL \times \text{cycle})</math>.</p> <p>When both SW write-leveling is enabled (i.e SW_WL_EN = 1) or HW write-leveling is enabled (i.e HW_WL_EN = 1 ) then this value will be taken as is and will be added to the associated delay that is configured in WL_DL_OFFSET and WL_HC_DEL.</p> <p>Note that in HW write-leveling this field is not used for indication, as in WL_DL_OFFSET and WL_HC_DEL, but for configuration.</p> <p>0 No delay is added. 1 1 cycle delay is added. 2 2 cycles delay is added. 3 Reserved.</p>
8 WL_HC_DELO	<p>Write leveling half cycle delay for Byte 0. This field indicates whether a delay of half cycle between CK and write DQS is added to the delay that is indicated in the associated WR_DL_ABS_OFFSET and WL_CYC_DEL. So the total delay is the sum of <math>(WL\_DL\_ABS\_OFFSET/256 \times \text{cycle}) + (WL\_HC\_DEL \times \text{half cycle}) + (WL\_CYC\_DEL \times \text{cycle})</math>.</p> <p>When SW write-leveling is enabled (i.e SW_WL_EN = 1) then this value will be taken as is and will be added to the associated delay that is configured in WL_DL_OFFSET and WL_CYC_DEL. When HW write-leveling is enabled (i.e HW_WL_EN = 1 ) then this value will indicate (status) whether a delay of half cycle was added or not to the associated WL_DL_OFFSET and WL_CYC_DEL.</p> <p>0 No delay is added. 1 Half cycle delay is added.</p>

*Table continues on the next page...*

**MMDC\_MPWLDECTRL0 field descriptions (continued)**

Field	Description
7 Reserved	This read-only field is reserved and always has the value 0.
6–0 WL_DL_ABS_OFFSET0	<p>Absolute write-leveling delay offset for Byte 0. This field indicates the absolute delay between CK and write DQS of Byte0 with fractions of a clock period and up to half cycle. This value is process and frequency independent. The value of the delay can be calculated using the following equation <math>(WR\_DL\_ABS\_OFFSET1 / 256) * \text{clock period}</math></p> <p>When SW write-leveling is enabled (i.e SW_WL_EN = 1) then this value will be taken as is to the associated delay-line. When HW write-leveling is enabled (i.e HW_WL_EN = 1 ) then this value will indicate (status) the value that is taken to the associated delay-line at the end of the write-leveling calibration.</p> <p><b>NOTE:</b> The delay-line has a resolution that may vary between device to device, therefore in some cases an increment of the delay by 1 step may be smaller than the delay-line resolution.</p>

**32.12.35 MMDC PHY Write Leveling Delay Control Register 1 (MMDC\_MPWLDECTRL1)**

Address: 21B\_0000h base + 810h offset = 21B\_0810h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					WL_CYC_DEL3		WL_HC_DEL3	0	WL_DL_ABS_OFFSET3						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					WL_CYC_DEL2		WL_HC_DEL2	0	WL_DL_ABS_OFFSET2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MPWLDECTRL1 field descriptions**

Field	Description
31–27 Reserved	This read-only field is reserved and always has the value 0.
26–25 WL_CYC_DEL3	Write leveling cycle delay for Byte 3. This field indicates whether a delay of 1 or 2 cycles between CK and write DQS is added to the delay that is indicated in the associated WL_DL_ABS_OFFSET and WL_HC_DEL. So the total delay is the sum of $(WL\_DL\_ABS\_OFFSET/256 * \text{cycle}) + (WL\_HC\_DEL * \text{half cycle}) + (WL\_CYC\_DEL * \text{cycle})$ .

*Table continues on the next page...*

## MMDC\_MPWLDECTRL1 field descriptions (continued)

Field	Description
	<p>When both SW write-leveling is enabled (i.e SW_WL_EN = 1) or HW write-leveling is enabled (i.e HW_WL_EN = 1 ) then this value will be taken as is and will be added to the associated delay that is configured in WL_DL_OFFSET and WL_HC_DEL.</p> <p>Note that in HW write-leveling this field is not used for indication, as in WL_DL_OFFSET and WL_HC_DEL, but for configuration.</p> <p>0 No delay is added.  1 1 cycle delay is added.  2 2 cycles delay is added.  3 Reserved.</p>
24 WL_HC_DEL3	<p>Write leveling half cycle delay for Byte 3. This field indicates whether a delay of half cycle between CK and write DQS is added to the delay that is indicated in the associated WL_DL_ABS_OFFSET and WL_CYC_DEL. So the total delay is the sum of (WL_DL_ABS_OFFSET/256*cycle) + (WL_HC_DEL*half cycle) + (WL_CYC_DEL*cycle).</p> <p>When SW write-leveling is enabled (i.e SW_WL_EN = 1) then this value will be taken as is and will be added to the associated delay that is configured in WL_DL_OFFSET and WL_CYC_DEL. When HW write-leveling is enabled (i.e HW_WL_EN = 1 ) then this value will indicate (status) whether a delay of half cycle was added or not to the associated WL_DL_OFFSET and WL_CYC_DEL.</p> <p>0 No delay is added.  1 Half cycle delay is added.</p>
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 WL_DL_ABS_OFFSET3	<p>Absolute write-leveling delay offset for Byte 3. This field indicates the absolute delay between CK and write DQS of Byte3 with fractions of a clock period and up to half cycle. This value is process and frequency independent. The value of the delay can be calculated using the following equation  <math display="block">(WL\_DL\_ABS\_OFFSET3 / 256) * \text{clock period}</math></p> <p>When SW write-leveling is enabled (i.e SW_WL_EN = 1) then this value will be taken as is to the associated delay-line. When HW write-leveling is enabled (i.e HW_WL_EN = 1 ) then this value will indicate (status) the value that is taken to the associated delay-line at the end of the write-leveling calibration.</p> <p><b>NOTE:</b> The delay-line has a resolution that may vary between device to device, therefore in some cases an increment of the delay by 1 step may be smaller than the delay-line resolution.</p>
15–11 Reserved	This read-only field is reserved and always has the value 0.
10–9 WL_CYC_DEL2	<p>Write leveling cycle delay for Byte 2. This field indicates whether a delay of 1 or 2 cycles between CK and write DQS is added to the delay that is indicated in the associated WL_DL_ABS_OFFSET and WL_HC_DEL. So the total delay is the sum of (WL_DL_ABS_OFFSET/256*cycle) + (WL_HC_DEL*half cycle) + (WL_CYC_DEL*cycle).</p> <p>When both SW write-leveling is enabled (i.e SW_WL_EN = 1) or HW write-leveling is enabled (i.e HW_WL_EN = 1 ) then this value will be taken as is and will be added to the associated delay that is configured in WL_DL_OFFSET and WL_HC_DEL.</p> <p>Note that in HW write-leveling this field is not used for indication, as in WL_DL_OFFSET and WL_HC_DEL, but for configuration.</p> <p>0 No delay is added.  1 1 cycle delay is added.  2 2 cycles delay is added.  3 Reserved.</p>

*Table continues on the next page...*

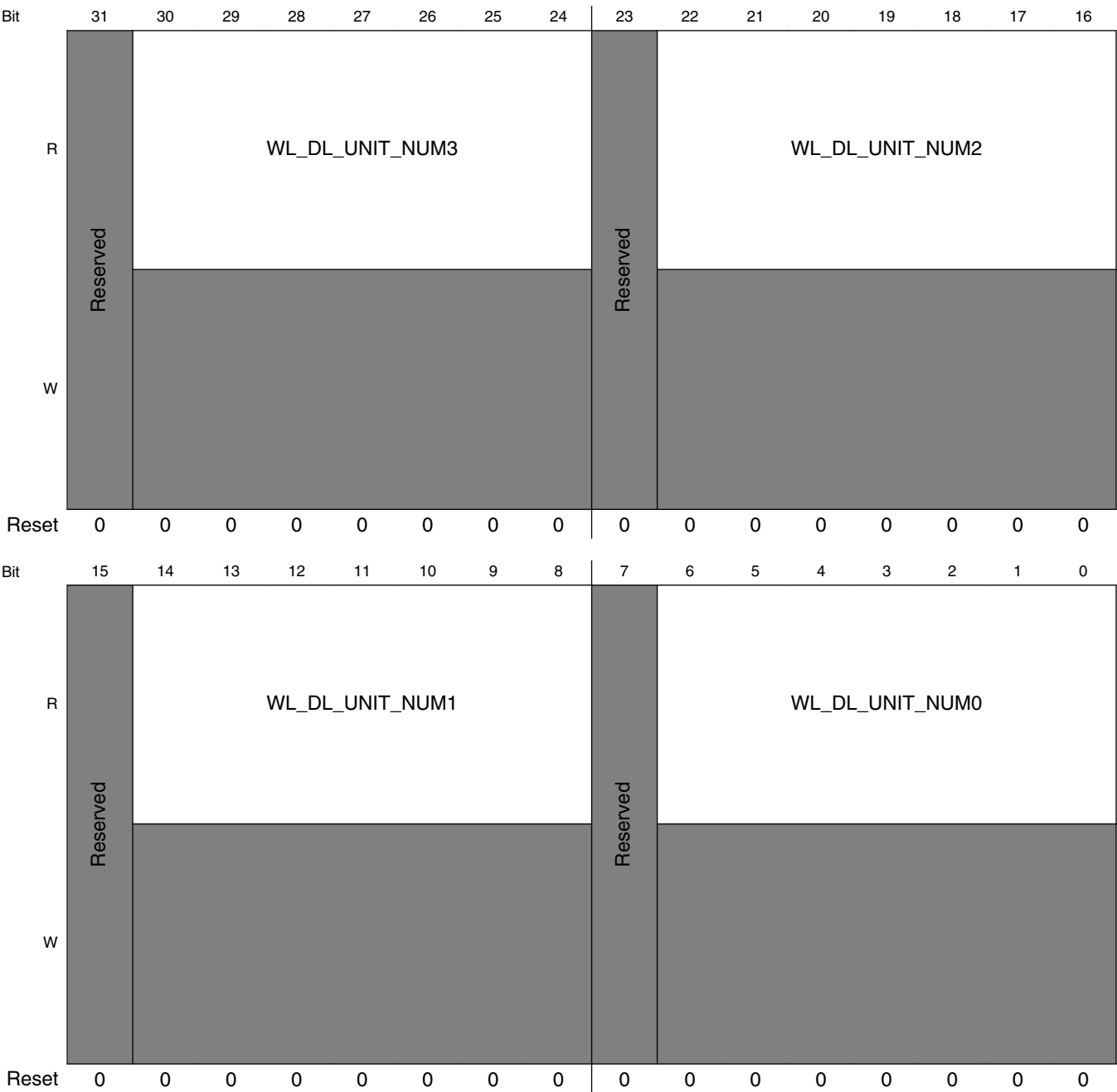
**MMDC\_MPWLDECTRL1 field descriptions (continued)**

Field	Description
8 WL_HC_DEL2	<p>Write leveling half cycle delay for Byte 2. This field indicates whether a delay of half cycle between CK and write DQS is added to the delay that is indicated in the associated WR_DL_ABS_OFFSET and WL_CYC_DEL. So the total delay is the sum of <math>(WL\_DL\_ABS\_OFFSET/256 \times \text{cycle}) + (WL\_HC\_DEL \times \text{half cycle}) + (WL\_CYC\_DEL \times \text{cycle})</math>.</p> <p>When SW write-leveling is enabled (i.e SW_WL_EN = 1) then this value will be taken as is and will be added to the associated delay that is configured in WL_DL_OFFSET and WL_CYC_DEL. When HW write-leveling is enabled (i.e HW_WL_EN = 1 ) then this value will indicate (status) whether a delay of half cycle was added or not to the associated WL_DL_OFFSET and WL_CYC_DEL.</p> <p>0 No delay is added. 1 Half cycle delay is added.</p>
7 Reserved	This read-only field is reserved and always has the value 0.
6-0 WL_DL_ABS_OFFSET2	<p>Absolute write-leveling delay offset for Byte 2. This field indicates the absolute delay between CK and write DQS of Byte1 with fractions of a clock period and up to half cycle. This value is process and frequency independent. The value of the delay can be calculated using the following equation <math>(WR\_DL\_ABS\_OFFSET2 / 256) \times \text{clock period}</math></p> <p>When SW write-leveling is enabled (i.e SW_WL_EN = 1) then this value will be taken as is to the associated delay-line. When HW write-leveling is enabled (i.e HW_WL_EN = 1 ) then this value will indicate (status) the value that is taken to the associated delay-line at the end of the write-leveling calibration.</p> <p><b>NOTE:</b> The delay-line has a resolution that may vary between device to device, therefore in some cases an increment of the delay by 1 step may be smaller than the delay-line resolution.</p>

### 32.12.36 MMDC PHY Write Leveling delay-line Status Register (MMDC\_MPWLDLST)

This register holds the status of the four write leveling delay-lines.

Address: 21B\_0000h base + 814h offset = 21B\_0814h





**MMDC\_MPWLDLST field descriptions**

<b>Field</b>	<b>Description</b>
31 -	This field is reserved. Reserved
30–24 WL_DL_UNIT_NUM3	This field reflects the number of delay units that are actually used by write leveling delay-line 3.
23 -	This field is reserved. Reserved
22–16 WL_DL_UNIT_NUM2	This field reflects the number of delay units that are actually used by write leveling delay-line 2.
15 -	This field is reserved. Reserved
14–8 WL_DL_UNIT_NUM1	This field reflects the number of delay units that are actually used by write leveling delay-line 1.
7 -	This field is reserved. Reserved
6–0 WL_DL_UNIT_NUM0	This field reflects the number of delay units that are actually used by write leveling delay-line 0.

## 32.12.37 MMDC PHY ODT control register (MMDC\_MPODTCTRL)

### NOTE

In LPDDR2 mode this register should be cleared, so no termination will be activated

Address: 21B\_0000h base + 818h offset = 21B\_0818h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													ODT3_INT_RES		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	ODT2_INT_RES				0	ODT1_INT_RES				0	ODT0_INT_RES				ODT_RD_ACT_EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
														</		

### MMDC\_MPODTCTRL field descriptions

Field	Description
31–19 Reserved	This read-only field is reserved and always has the value 0.
18–16 ODT3_INT_RES	On chip ODT byte3 resistor - This field determines the Rtt_Nom of the on chip ODT byte3 resistor during read accesses.  000 Rtt_Nom Disabled. 001 Rtt_Nom 120 Ohm 010 Rtt_Nom 60 Ohm 011 Rtt_Nom 40 Ohm 100 Rtt_Nom 30 Ohm 101 Rtt_Nom 24 Ohm 110 Rtt_Nom 20 Ohm 111 Rtt_Nom 17 Ohm
15 Reserved	This read-only field is reserved and always has the value 0.
14–12 ODT2_INT_RES	On chip ODT byte2 resistor - This field determines the Rtt_Nom of the on chip ODT byte2 resistor during read accesses.  000 Rtt_Nom Disabled.

Table continues on the next page...

**MMDC\_MPODTCTRL field descriptions (continued)**

Field	Description
	001 Rtt_Nom 120 Ohm 010 Rtt_Nom 60 Ohm 011 Rtt_Nom 40 Ohm 100 Rtt_Nom 30 Ohm 101 Rtt_Nom 24 Ohm 110 Rtt_Nom 20 Ohm 111 Rtt_Nom 17 Ohm
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 ODT1_INT_RES	On chip ODT byte1 resistor - This field determines the Rtt_Nom of the on chip ODT byte1 resistor during read accesses.  0000 Rtt_Nom Disabled. 001 Rtt_Nom 120 Ohm 010 Rtt_Nom 60 Ohm 011 Rtt_Nom 40 Ohm 100 Rtt_Nom 30 Ohm 101 Rtt_Nom 24 Ohm 110 Rtt_Nom 20 Ohm 111 Rtt_Nom 17 Ohm
7 Reserved	This read-only field is reserved and always has the value 0.
6–4 ODT0_INT_RES	On chip ODT byte0 resistor - This field determines the Rtt_Nom of the on chip ODT byte0 resistor during read accesses.  000 Rtt_Nom Disabled. 001 Rtt_Nom 120 Ohm 010 Rtt_Nom 60 Ohm 011 Rtt_Nom 40 Ohm 100 Rtt_Nom 30 Ohm 101 Rtt_Nom 24 Ohm 110 Rtt_Nom 20 Ohm 111 Rtt_Nom 17 Ohm
3 ODT_RD_ACT_EN	Active read CS ODT enable. The bit determines if ODT pin of the active CS will be asserted during read accesses.  0 Active CS ODT pin is disabled during read access. 1 Active CS ODT pin is enabled during read access.
2 ODT_RD_PAS_EN	Inactive read CS ODT enable. The bit determines if ODT pin of the inactive CS will be asserted during read accesses.  0 Inactive CS ODT pin is disabled during read accesses to other CS. 1 Inactive CS ODT pin is enabled during read accesses to other CS.
1 ODT_WR_ACT_EN	Active write CS ODT enable. The bit determines if ODT pin of the active CS will be asserted during write accesses.  0 Active CS ODT pin is disabled during write access. 1 Active CS ODT pin is enabled during write access.

*Table continues on the next page...*

**MMDC\_MPODTCTRL field descriptions (continued)**

Field	Description
0 ODT_WR_PAS_EN	Inactive write CS ODT enable. The bit determines if ODT pin of the inactive CS will be asserted during write accesses.  0 Inactive CS ODT pin is disabled during write accesses to other CS. 1 Inactive CS ODT pin is enabled during write accesses to other CS.

**32.12.38 MMDC PHY Read DQ Byte0 Delay Register (MMDC\_MPRDDQBY0DL)**

This register is used to add fine-tuning adjustment to every bit in the read DQ byte0 relative to the read DQS. This delay is in addition to the read data calibration. If operating in 64-bit mode, there is an identical register that is mapped at the second base address.

Address: 21B\_0000h base + 81Ch offset = 21B\_081Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16				
R	0	rd_dq7_del				0	rd_dq6_del				0	rd_dq5_del				0	rd_dq4_del			
W																				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
R	0	rd_dq3_del				0	rd_dq2_del				0	rd_dq1_del				0	rd_dq0_del			
W																				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

**MMDC\_MPRDDQBY0DL field descriptions**

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–28 rd_dq7_del	Read dqs0 to dq7 delay fine-tuning. This field holds the number of delay units that are added to dq7 relative to dqs0.  000 No change in dq7 delay 001 Add dq7 delay of 1 delay unit 010 Add dq7 delay of 2 delay units. 011 Add dq7 delay of 3 delay units. 100 Add dq7 delay of 4 delay units. 101 Add dq7 delay of 5 delay units. 110 Add dq7 delay of 6 delay units. 111 Add dq7 delay of 7 delay units.
27 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**MMDC\_MPRDDQBY0DL field descriptions (continued)**

Field	Description
26–24 rd_dq6_del	Read dqs0 to dq6 delay fine-tuning. This field holds the number of delay units that are added to dq6 relative to dqs0.  000 No change in dq6 delay 001 Add dq6 delay of 1 delay unit 010 Add dq6 delay of 2 delay units. 011 Add dq6 delay of 3 delay units. 100 Add dq6 delay of 4 delay units. 101 Add dq6 delay of 5 delay units. 110 Add dq6 delay of 6 delay units. 111 Add dq6 delay of 7 delay units.
23 Reserved	This read-only field is reserved and always has the value 0.
22–20 rd_dq5_del	Read dqs0 to dq5 delay fine-tuning. This field holds the number of delay units that are added to dq5 relative to dqs0.  000 No change in dq5 delay 001 Add dq5 delay of 1 delay unit 010 Add dq5 delay of 2 delay units. 011 Add dq5 delay of 3 delay units. 100 Add dq5 delay of 4 delay units. 101 Add dq5 delay of 5 delay units. 110 Add dq5 delay of 6 delay units. 111 Add dq5 delay of 7 delay units.
19 Reserved	This read-only field is reserved and always has the value 0.
18–16 rd_dq4_del	Read dqs0 to dq4 delay fine-tuning. This field holds the number of delay units that are added to dq4 relative to dqs0.  000 No change in dq4 delay 001 Add dq4 delay of 1 delay unit 010 Add dq4 delay of 2 delay units. 011 Add dq4 delay of 3 delay units. 100 Add dq4 delay of 4 delay units. 101 Add dq4 delay of 5 delay units. 110 Add dq4 delay of 6 delay units. 111 Add dq4 delay of 7 delay units.
15 Reserved	This read-only field is reserved and always has the value 0.
14–12 rd_dq3_del	Read dqs0 to dq3 delay fine-tuning. This field holds the number of delay units that are added to dq3 relative to dqs0.  000 No change in dq3 delay 001 Add dq3 delay of 1 delay unit 010 Add dq3 delay of 2 delay units. 011 Add dq3 delay of 3 delay units. 100 Add dq3 delay of 4 delay units. 101 Add dq3 delay of 5 delay units.

*Table continues on the next page...*

### MMDC\_MPRDDQBY0DL field descriptions (continued)

Field	Description
	110 Add dq3 delay of 6 delay units. 111 Add dq3 delay of 7 delay units.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 rd_dq2_del	Read dqs0 to dq2 delay fine-tuning. This field holds the number of delay units that are added to dq2 relative to dqs0.  000 No change in dq2 delay 001 Add dq2 delay of 1 delay unit 010 Add dq2 delay of 2 delay units. 011 Add dq2 delay of 3 delay units. 100 Add dq2 delay of 4 delay units. 101 Add dq2 delay of 5 delay units. 110 Add dq2 delay of 6 delay units. 111 Add dq2 delay of 7 delay units.
7 Reserved	This read-only field is reserved and always has the value 0.
6–4 rd_dq1_del	Read dqs0 to dq1 delay fine-tuning. This field holds the number of delay units that are added to dq1 relative to dqs0.  000 No change in dq1 delay 001 Add dq1 delay of 1 delay unit 010 Add dq1 delay of 2 delay units. 011 Add dq1 delay of 3 delay units. 100 Add dq1 delay of 4 delay units. 101 Add dq1 delay of 5 delay units. 110 Add dq1 delay of 6 delay units. 111 Add dq1 delay of 7 delay units.
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 rd_dq0_del	Read dqs0 to dq0 delay fine-tuning. This field holds the number of delay units that are added to dq0 relative to dqs0.  000 No change in dq0 delay 001 Add dq0 delay of 1 delay unit 010 Add dq0 delay of 2 delay units. 011 Add dq0 delay of 3 delay units. 100 Add dq0 delay of 4 delay units. 101 Add dq0 delay of 5 delay units. 110 Add dq0 delay of 6 delay units. 111 Add dq0 delay of 7 delay units.

### 32.12.39 MMDC PHY Read DQ Byte1 Delay Register (MMDC\_MPRDDQBY1DL)

This register is used to add fine-tuning adjustment to every bit in the read DQ byte1 relative to the read DQS

Address: 21B\_0000h base + 820h offset = 21B\_0820h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16				
R	0	rd_dq15_del				0	rd_dq14_del				0	rd_dq13_del				0	rd_dq12_del			
W																				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
R	0	rd_dq11_del				0	rd_dq10_del				0	rd_dq9_del				0	rd_dq8_del			
W																				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

#### MMDC\_MPRDDQBY1DL field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–28 rd_dq15_del	Read dqs1 to dq15 delay fine-tuning. This field holds the number of delay units that are added to dq15 relative to dqs1.  000 No change in dq15 delay 001 Add dq15 delay of 1 delay unit 010 Add dq15 delay of 2 delay units. 011 Add dq15 delay of 3 delay units. 100 Add dq15 delay of 4 delay units. 101 Add dq15 delay of 5 delay units. 110 Add dq15 delay of 6 delay units. 111 Add dq15 delay of 7 delay units.
27 Reserved	This read-only field is reserved and always has the value 0.
26–24 rd_dq14_del	Read dqs1 to dq14 delay fine-tuning. This field holds the number of delay units that are added to dq14 relative to dqs1.  000 No change in dq14 delay 001 Add dq14 delay of 1 delay unit 010 Add dq14 delay of 2 delay units. 011 Add dq14 delay of 3 delay units. 100 Add dq14 delay of 4 delay units. 101 Add dq14 delay of 5 delay units. 110 Add dq14 delay of 6 delay units. 111 Add dq14 delay of 7 delay units.

Table continues on the next page...

### MMDC\_MPRDDQBY1DL field descriptions (continued)

Field	Description
23 Reserved	This read-only field is reserved and always has the value 0.
22–20 rd_dq13_del	Read dqs1 to dq13 delay fine-tuning. This field holds the number of delay units that are added to dq13 relative to dqs1.  000 No change in dq13 delay 001 Add dq13 delay of 1 delay unit 010 Add dq13 delay of 2 delay units. 011 Add dq13 delay of 3 delay units. 100 Add dq13 delay of 4 delay units. 101 Add dq13 delay of 5 delay units. 110 Add dq13 delay of 6 delay units. 111 Add dq13 delay of 7 delay units.
19 Reserved	This read-only field is reserved and always has the value 0.
18–16 rd_dq12_del	Read dqs1 to dq12 delay fine-tuning. This field holds the number of delay units that are added to dq12 relative to dqs1.  000 No change in dq12 delay 001 Add dq12 delay of 1 delay unit 010 Add dq12 delay of 2 delay units. 011 Add dq12 delay of 3 delay units. 100 Add dq12 delay of 4 delay units. 101 Add dq12 delay of 5 delay units. 110 Add dq12 delay of 6 delay units. 111 Add dq12 delay of 7 delay units.
15 Reserved	This read-only field is reserved and always has the value 0.
14–12 rd_dq11_del	Read dqs1 to dq11 delay fine-tuning. This field holds the number of delay units that are added to dq11 relative to dqs1.  000 No change in dq11 delay 001 Add dq11 delay of 1 delay unit 010 Add dq11 delay of 2 delay units. 011 Add dq11 delay of 3 delay units. 100 Add dq11 delay of 4 delay units. 101 Add dq11 delay of 5 delay units. 110 Add dq11 delay of 6 delay units. 111 Add dq11 delay of 7 delay units.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 rd_dq10_del	Read dqs1 to dq10 delay fine-tuning. This field holds the number of delay units that are added to dq10 relative to dqs1.  000 No change in dq10 delay 001 Add dq10 delay of 1 delay unit 010 Add dq10 delay of 2 delay units. 011 Add dq10 delay of 3 delay units.

Table continues on the next page...



**MMDC\_MPRDDQBY1DL field descriptions (continued)**

Field	Description
	100 Add dq10 delay of 4 delay units. 101 Add dq10 delay of 5 delay unit 110 Add dq10 delay of 6 delay units. 111 Add dq10 delay of 7 delay units.
7 Reserved	This read-only field is reserved and always has the value 0.
6–4 rd_dq9_del	Read dqs1 to dq9 delay fine-tuning. This field holds the number of delay units that are added to dq9 relative to dqs1.  000 No change in dq9 delay 001 Add dq9 delay of 1 delay unit 010 Add dq9 delay of 2 delay units. 011 Add dq9 delay of 3 delay units. 100 Add dq9 delay of 4 delay units. 101 Add dq9 delay of 5 delay units. 110 Add dq9 delay of 6 delay units. 111 Add dq9 delay of 7 delay units.
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 rd_dq8_del	Read dqs1 to dq8 delay fine-tuning. This field holds the number of delay units that are added to dq8 relative to dqs1.  000 No change in dq8 delay 001 Add dq8 delay of 1 delay unit 010 Add dq8 delay of 2 delay units. 011 Add dq8 delay of 3 delay units. 100 Add dq8 delay of 4 delay units. 101 Add dq8 delay of 5 delay units. 110 Add dq8 delay of 6 delay units. 111 Add dq8 delay of 7 delay units.

## 32.12.40 MMDC PHY Read DQ Byte2 Delay Register (MMDC\_MPRDDQBY2DL)

This register is used to add fine-tuning adjustment to every bit in the read DQ byte2 relative to the read DQS

Address: 21B\_0000h base + 824h offset = 21B\_0824h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16				
R	0	rd_dq23_del				0	rd_dq22_del				0	rd_dq21_del				0	rd_dq20_del			
W																				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
R	0	rd_dq19_del				0	rd_dq18_del				0	rd_dq17_del				0	rd_dq16_del			
W																				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

### MMDC\_MPRDDQBY2DL field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–28 rd_dq23_del	Read dqs2 to dq23 delay fine-tuning. This field holds the number of delay units that are added to dq23 relative to dqs2.  000 No change in dq23 delay 001 Add dq23 delay of 1 delay unit 010 Add dq23 delay of 2 delay units. 011 Add dq23 delay of 3 delay units. 100 Add dq23 delay of 4 delay units. 101 Add dq23 delay of 5 delay units. 110 Add dq23 delay of 6 delay units. 111 Add dq23 delay of 7 delay units.
27 Reserved	This read-only field is reserved and always has the value 0.
26–24 rd_dq22_del	Read dqs2 to dq22 delay fine-tuning. This field holds the number of delay units that are added to dq22 relative to dqs2.  000 No change in dq22 delay 001 Add dq22 delay of 1 delay unit 010 Add dq22 delay of 2 delay units. 011 Add dq22 delay of 3 delay units. 100 Add dq22 delay of 4 delay units. 101 Add dq22 delay of 5 delay units. 110 Add dq22 delay of 6 delay units. 111 Add dq22 delay of 7 delay units.

Table continues on the next page...

**MMDC\_MPRDDQBY2DL field descriptions (continued)**

Field	Description
23 Reserved	This read-only field is reserved and always has the value 0.
22–20 rd_dq21_del	Read dqs2 to dq21 delay fine-tuning. This field holds the number of delay units that are added to dq21 relative to dqs2.  000 No change in dq21 delay 001 Add dq21 delay of 1 delay unit 010 Add dq21 delay of 2 delay units. 011 Add dq21 delay of 3 delay units. 100 Add dq21 delay of 4 delay units. 101 Add dq21 delay of 5 delay units. 110 Add dq21 delay of 6 delay units. 111 Add dq21 delay of 7 delay units.
19 Reserved	This read-only field is reserved and always has the value 0.
18–16 rd_dq20_del	Read dqs2 to dq20 delay fine-tuning. This field holds the number of delay units that are added to dq20 relative to dqs2.  000 No change in dq20 delay 001 Add dq20 delay of 1 delay unit 010 Add dq20 delay of 2 delay units. 011 Add dq20 delay of 3 delay units. 100 Add dq20 delay of 4 delay units. 101 Add dq20 delay of 5 delay units. 110 Add dq20 delay of 6 delay units. 111 Add dq20 delay of 7 delay units.
15 Reserved	This read-only field is reserved and always has the value 0.
14–12 rd_dq19_del	Read dqs2 to dq19 delay fine-tuning. This field holds the number of delay units that are added to dq19 relative to dqs2.  000 No change in dq19 delay 001 Add dq19 delay of 1 delay unit 010 Add dq19 delay of 2 delay units. 011 Add dq19 delay of 3 delay units. 100 Add dq19 delay of 4 delay units. 101 Add dq19 delay of 5 delay units. 110 Add dq19 delay of 6 delay units. 111 Add dq19 delay of 7 delay units.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 rd_dq18_del	Read dqs2 to dq18 delay fine-tuning. This field holds the number of delay units that are added to dq18 relative to dqs2.  000 No change in dq18 delay 001 Add dq18 delay of 1 delay unit 010 Add dq18 delay of 2 delay units. 011 Add dq18 delay of 3 delay units.

*Table continues on the next page...*

### MMDC\_MPRDDQBY2DL field descriptions (continued)

Field	Description
	100 Add dq18 delay of 4 delay units. 101 Add dq18 delay of 5 delay units. 110 Add dq18 delay of 6 delay units. 111 Add dq18 delay of 7 delay units.
7 Reserved	This read-only field is reserved and always has the value 0.
6–4 rd_dq17_del	Read dqs2 to dq17 delay fine-tuning. This field holds the number of delay units that are added to dq17 relative to dqs2.  000 No change in dq17 delay 001 Add dq17 delay of 1 delay unit 010 Add dq17 delay of 2 delay units. 011 Add dq17 delay of 3 delay units. 100 Add dq17 delay of 4 delay units. 101 Add dq17 delay of 5 delay units. 110 Add dq17 delay of 6 delay units. 111 Add dq17 delay of 7 delay units.
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 rd_dq16_del	Read dqs2 to dq16 delay fine-tuning. This field holds the number of delay units that are added to dq16 relative to dqs2.  000 No change in dq16 delay 001 Add dq16 delay of 1 delay unit 010 Add dq16 delay of 2 delay units. 011 Add dq16 delay of 3 delay units. 100 Add dq16 delay of 4 delay units. 101 Add dq16 delay of 5 delay units. 110 Add dq16 delay of 6 delay units. 111 Add dq16 delay of 7 delay units.

### 32.12.41 MMDC PHY Read DQ Byte3 Delay Register (MMDC\_MPRDDQBY3DL)

This register is used to add fine-tuning adjustment to every bit in the read DQ byte3 relative to the read DQS.

The bit assignments and the bit field descriptions for the register are shown below.

Address: 21B\_0000h base + 828h offset = 21B\_0828h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16				
R	0	rd_dq31_del				0	rd_dq30_del				0	rd_dq29_del				0	rd_dq28_del			
W																				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0				

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	rd_dq27_del				0	rd_dq26_del			0	rd_dq25_del			0	rd_dq24_del	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MPRDDQBY3DL field descriptions**

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–28 rd_dq31_del	Read dqs3 to dq31 delay fine-tuning. This field holds the number of delay units that are added to dq31 relative to dqs3.  000 No change in dq31 delay 001 Add dq31 delay of 1 delay unit 010 Add dq31 delay of 2 delay units. 011 Add dq31 delay of 3 delay units. 100 Add dq31 delay of 4 delay units. 101 Add dq31 delay of 5 delay units. 110 Add dq31 delay of 6 delay units. 111 Add dq31 delay of 7 delay units.
27 Reserved	This read-only field is reserved and always has the value 0.
26–24 rd_dq30_del	Read dqs3 to dq30 delay fine-tuning. This field holds the number of delay units that are added to dq30 relative to dqs3.  000 No change in dq30 delay 001 Add dq30 delay of 1 delay unit 010 Add dq30 delay of 2 delay units. 011 Add dq30 delay of 3 delay units. 100 Add dq30 delay of 4 delay units. 101 Add dq30 delay of 5 delay units. 110 Add dq30 delay of 6 delay units. 111 Add dq30 delay of 7 delay units.
23 Reserved	This read-only field is reserved and always has the value 0.
22–20 rd_dq29_del	Read dqs3 to dq29 delay fine-tuning. This field holds the number of delay units that are added to dq29 relative to dqs3.  000 No change in dq29 delay 001 Add dq29 delay of 1 delay unit 010 Add dq29 delay of 2 delay units. 011 Add dq29 delay of 3 delay units. 100 Add dq29 delay of 4 delay units. 101 Add dq29 delay of 5 delay units. 110 Add dq29 delay of 6 delay units. 111 Add dq29 delay of 7 delay units.
19 Reserved	This read-only field is reserved and always has the value 0.

*Table continues on the next page...*

### MMDC\_MPRDDQBY3DL field descriptions (continued)

Field	Description
18–16 rd_dq28_del	Read dqs3 to dq28 delay fine-tuning. This field holds the number of delay units that are added to dq28 relative to dqs3.  000 No change in dq28 delay 001 Add dq28 delay of 1 delay unit 010 Add dq28 delay of 2 delay units. 011 Add dq28 delay of 3 delay units. 100 Add dq28 delay of 4 delay units. 101 Add dq28 delay of 5 delay units. 110 Add dq28 delay of 6 delay units. 111 Add dq28 delay of 7 delay units.
15 Reserved	This read-only field is reserved and always has the value 0.
14–12 rd_dq27_del	Read dqs3 to dq27 delay fine-tuning. This field holds the number of delay units that are added to dq27 relative to dqs3.  000 No change in dq27 delay 001 Add dq27 delay of 1 delay unit 010 Add dq27 delay of 2 delay units. 011 Add dq27 delay of 3 delay units. 100 Add dq27 delay of 4 delay units. 101 Add dq27 delay of 5 delay units. 110 Add dq27 delay of 6 delay units. 111 Add dq27 delay of 7 delay units.
11 Reserved	This read-only field is reserved and always has the value 0.
10–8 rd_dq26_del	Read dqs3 to dq26 delay fine-tuning. This field holds the number of delay units that are added to dq26 relative to dqs3.  000 No change in dq26 delay 001 Add dq26 delay of 1 delay unit 010 Add dq26 delay of 2 delay units. 011 Add dq26 delay of 3 delay units. 100 Add dq26 delay of 4 delay units. 101 Add dq26 delay of 5 delay units. 110 Add dq26 delay of 6 delay units. 111 Add dq26 delay of 7 delay units.
7 Reserved	This read-only field is reserved and always has the value 0.
6–4 rd_dq25_del	Read dqs3 to dq25 delay fine-tuning. This field holds the number of delay units that are added to dq25 relative to dqs3.  000 No change in dq25 delay 001 Add dq25 delay of 1 delay unit 010 Add dq25 delay of 2 delay units. 011 Add dq25 delay of 3 delay units. 100 Add dq25 delay of 4 delay units. 101 Add dq25 delay of 5 delay units.

Table continues on the next page...

**MMDC\_MPRDDQBY3DL field descriptions (continued)**

Field	Description
	110 Add dq25 delay of 6 delay units. 111 Add dq25 delay of 7 delay units.
3 Reserved	This read-only field is reserved and always has the value 0.
2–0 rd_dq24_del	Read dqs3 to dq24 delay fine-tuning. This field holds the number of delay units that are added to dq24 relative to dqs3.  000 No change in dq24 delay 001 Add dq24 delay of 1 delay unit 010 Add dq24 delay of 2 delay units. 011 Add dq24 delay of 3 delay units. 100 Add dq24 delay of 4 delay units. 101 Add dq24 delay of 5 delay units. 110 Add dq24 delay of 6 delay units. 111 Add dq24 delay of 7 delay units.

**32.12.42 MMDC PHY Write DQ Byte0 Delay Register (MMDC\_MPWRDQBY0DL)**

This register is used to add fine-tuning adjustment to every bit in the write DQ byte0 relative to the write DQS

Address: 21B\_0000h base + 82Ch offset = 21B\_082Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MPWRDQBY0DL field descriptions**

Field	Description
31–30 wr_dm0_del	Write dm0 delay fine-tuning. This field holds the number of delay units that are added to dm0 relative to dqs0.  00 No change in dm0 delay 01 Add dm0 delay of 1 delay unit. 10 Add dm0 delay of 2 delay units. 11 Add dm0 delay of 3 delay units.

*Table continues on the next page...*

### MMDC\_MPWRDQBY0DL field descriptions (continued)

Field	Description
29–28 wr_dq7_del	Write dq7 delay fine-tuning. This field holds the number of delay units that are added to dq7 relative to dqs0.  00 No change in dq7 delay 01 Add dq7 delay of 1 delay unit. 10 Add dq7 delay of 2 delay units. 11 Add dq7 delay of 3 delay units.
27–26 Reserved	This read-only field is reserved and always has the value 0.
25–24 wr_dq6_del	Write dq6 delay fine-tuning. This field holds the number of delay units that are added to dq6 relative to dqs0.  00 No change in dq6 delay 01 Add dq6 delay of 1 delay unit. 10 Add dq6 delay of 2 delay units. 11 Add dq6 delay of 3 delay units.
23–22 Reserved	This read-only field is reserved and always has the value 0.
21–20 wr_dq5_del	Write dq5 delay fine-tuning. This field holds the number of delay units that are added to dq5 relative to dqs0.  00 No change in dq5 delay 01 Add dq5 delay of 1 delay unit. 10 Add dq5 delay of 2 delay units. 11 Add dq5 delay of 3 delay units.
19–18 Reserved	This read-only field is reserved and always has the value 0.
17–16 wr_dq4_del	Write dq4 delay fine-tuning. This field holds the number of delay units that are added to dq4 relative to dqs0.  00 No change in dq4 delay 01 Add dq4 delay of 1 delay unit.. 10 Add dq4 delay of 2 delay units. 11 Add dq4 delay of 3 delay units.
15–14 Reserved	This read-only field is reserved and always has the value 0.
13–12 wr_dq3_del	Write dq3 delay fine-tuning. This field holds the number of delay units that are added to dq3 relative to dqs0.  00 No change in dq3 delay 01 Add dq3 delay of 1 delay unit. 10 Add dq3 delay of 2 delay units. 11 Add dq3 delay of 3 delay units.
11–10 Reserved	This read-only field is reserved and always has the value 0.
9–8 wr_dq2_del	Write dq2 delay fine-tuning. This field holds the number of delay units that are added to dq2 relative to dqs0.  00 No change in dq2 delay

Table continues on the next page...



**MMDC\_MPWRDQBY0DL field descriptions (continued)**

Field	Description
	01 Add dq2 delay of 1 delay unit. 10 Add dq2 delay of 2 delay units. 11 Add dq2 delay of 3 delay units.
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–4 wr_dq1_del	Write dq1 delay fine-tuning. This field holds the number of delay units that are added to dq1 relative to dqs0.  00 No change in dq1 delay 01 Add dq1 delay of 1 delay unit. 10 Add dq1 delay of 2 delay units. 11 Add dq1 delay of 3 delay units.
3–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 wr_dq0_del	Write dq0 delay fine-tuning. This field holds the number of delay units that are added to dq0 relative to dqs0.  00 No change in dq0 delay 01 Add dq0 delay of 1 delay unit. 10 Add dq0 delay of 2 delay units. 11 Add dq0 delay of 3 delay units.

**32.12.43 MMDC PHY Write DQ Byte1 Delay Register (MMDC\_MPWRDQBY1DL)**

This register is used to add fine-tuning adjustment to every bit in the write DQ byte1 relative to the write DQS

Address: 21B\_0000h base + 830h offset = 21B\_0830h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R						0				0				0		
W	wr_dm1_del	wr_dq15_del					wr_dq14_del				wr_dq13_del				wr_dq12_del	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0					0				0		
W		wr_dq11_del					wr_dq10_del				wr_dq9_del				wr_dq8_del	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MPWRDQBY1DL field descriptions**

Field	Description
31–30 wr_dm1_del	Write dm1 delay fine-tuning. This field holds the number of delay units that are added to dm1 relative to dqs1.

Table continues on the next page...

### MMDC\_MPWRDQBY1DL field descriptions (continued)

Field	Description
	00 No change in dm1 delay 01 Add dm1 delay of 1 delay unit. 10 Add dm1 delay of 2 delay units. 11 Add dm1 delay of 3 delay units.
29–28 wr_dq15_del	Write dq15 delay fine-tuning. This field holds the number of delay units that are added to dq15 relative to dqs1.  00 No change in dq15 delay 01 Add dq15 delay of 1 delay unit. 10 Add dq15 delay of 2 delay units. 11 Add dq15 delay of 3 delay units.
27–26 Reserved	This read-only field is reserved and always has the value 0.
25–24 wr_dq14_del	Write dq14 delay fine-tuning. This field holds the number of delay units that are added to dq14 relative to dqs1.  00 No change in dq14 delay 01 Add dq14 delay of 1 delay unit. 10 Add dq14 delay of 2 delay units. 11 Add dq14 delay of 3 delay units.
23–22 Reserved	This read-only field is reserved and always has the value 0.
21–20 wr_dq13_del	Write dq13 delay fine-tuning. This field holds the number of delay units that are added to dq13 relative to dqs1.  00 No change in dq13 delay 01 Add dq13 delay of 1 delay unit. 10 Add dq13 delay of 2 delay units. 11 Add dq13 delay of 3 delay units.
19–18 Reserved	This read-only field is reserved and always has the value 0.
17–16 wr_dq12_del	Write dq12 delay fine-tuning. This field holds the number of delay units that are added to dq12 relative to dqs1.  00 No change in dq12 delay 01 Add dq12 delay of 1 delay unit. 10 Add dq12 delay of 2 delay units. 11 Add dq12 delay of 3 delay units.
15–14 Reserved	This read-only field is reserved and always has the value 0.
13–12 wr_dq11_del	Write dq11 delay fine-tuning. This field holds the number of delay units that are added to dq11 relative to dqs1.  00 No change in dq11 delay 01 Add dq11 delay of 1 delay unit. 10 Add dq11 delay of 2 delay units. 11 Add dq11 delay of 3 delay units.

Table continues on the next page...

**MMDC\_MPWRDQBY1DL field descriptions (continued)**

Field	Description
11–10 Reserved	This read-only field is reserved and always has the value 0.
9–8 wr_dq10_del	Write dq10 delay fine-tuning. This field holds the number of delay units that are added to dq10 relative to dqs1.  00 No change in dq10 delay 01 Add dq10 delay of 1 delay unit. 10 Add dq10 delay of 2 delay units. 11 Add dq10 delay of 3 delay units.
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–4 wr_dq9_del	Write dq9 delay fine-tuning. This field holds the number of delay units that are added to dq9 relative to dqs1.  00 No change in dq9 delay 01 Add dq9 delay of 1 delay unit. 10 Add dq9 delay of 2 delay units. 11 Add dq9 delay of 3 delay units.
3–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 wr_dq8_del	Write dq8 delay fine-tuning. This field holds the number of delay units that are added to dq8 relative to dqs1.  00 No change in dq8 delay 01 Add dq8 delay of 1 delay unit. 10 Add dq8 delay of 2 delay units. 11 Add dq8 delay of 3 delay units.

**32.12.44 MMDC PHY Write DQ Byte2 Delay Register (MMDC\_MPWRDQBY2DL)**

This register is used to add fine-tuning adjustment to every bit in the write DQ byte2 relative to the write DQS

Address: 21B\_0000h base + 834h offset = 21B\_0834h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R						0				0				0		
W	wr_dm2_del		wr_dq23_del				wr_dq22_del				wr_dq21_del				wr_dq20_del	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R		0				0				0				0		
W			wr_dq19_del				wr_dq18_del				wr_dq17_del				wr_dq16_del	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### MMDC\_MPWRDQBY2DL field descriptions

Field	Description
31–30 wr_dm2_del	Write dm2 delay fine-tuning. This field holds the number of delay units that are added to dm2 relative to dqs2.  00 No change in dm2 delay 01 Add dm2 delay of 1 delay unit. 10 Add dm2 delay of 2 delay units. 11 Add dm2 delay of 3 delay units.
29–28 wr_dq23_del	Write dq23 delay fine tuning. This field holds the number of delay units that are added to dq23 relative to dqs2.  00 No change in dq23 delay 01 Add dq23 delay of 1 delay unit. 10 Add dq23 delay of 2 delay units. 11 Add dq23 delay of 3 delay units.
27–26 Reserved	This read-only field is reserved and always has the value 0.
25–24 wr_dq22_del	Write dq22 delay fine tuning. This field holds the number of delay units that are added to dq22 relative to dqs2.  00 No change in dq22 delay 01 Add dq22 delay of 1 delay unit. 10 Add dq22 delay of 2 delay units. 11 Add dq22 delay of 3 delay units.
23–22 Reserved	This read-only field is reserved and always has the value 0.
21–20 wr_dq21_del	Write dq21 delay fine tuning. This field holds the number of delay units that are added to dq21 relative to dqs2.  00 No change in dq21 delay 01 Add dq21 delay of 1 delay unit. 10 Add dq21 delay of 2 delay units. 11 Add dq21 delay of 3 delay units.
19–18 Reserved	This read-only field is reserved and always has the value 0.
17–16 wr_dq20_del	Write dq20 delay fine tuning. This field holds the number of delay units that are added to dq20 relative to dqs2.  00 No change in dq20 delay 01 Add dq20 delay of 1 delay unit. 10 Add dq20 delay of 2 delay units. 11 Add dq20 delay of 3 delay units.
15–14 Reserved	This read-only field is reserved and always has the value 0.
13–12 wr_dq19_del	Write dq19 delay fine tuning. This field holds the number of delay units that are added to dq19 relative to dqs2.  00 No change in dq19 delay 01 Add dq19 delay of 1 delay unit.

Table continues on the next page...

**MMDC\_MPWRDQBY2DL field descriptions (continued)**

Field	Description
	10 Add dq19 delay of 2 delay units. 11 Add dq19 delay of 3 delay units.
11–10 Reserved	This read-only field is reserved and always has the value 0.
9–8 wr_dq18_del	Write dq18 delay fine tuning. This field holds the number of delay units that are added to dq18 relative to dqs2.  00 No change in dq18 delay 01 Add dq18 delay of 1 delay unit. 10 Add dq18 delay of 2 delay units. 11 Add dq18 delay of 3 delay units.
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–4 wr_dq17_del	Write dq17 delay fine tuning. This field holds the number of delay units that are added to dq17 relative to dqs2.  00 No change in dq17 delay 01 Add dq17 delay of 1 delay unit. 10 Add dq17 delay of 2 delay units. 11 Add dq17 delay of 3 delay units.
3–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 wr_dq16_del	Write dq16 delay fine tuning. This field holds the number of delay units that are added to dq16 relative to dqs2.  00 No change in dq16 delay 01 Add dq16 delay of 1 delay unit. 10 Add dq16 delay of 2 delay units. 11 Add dq16 delay of 3 delay units.

## 32.12.45 MMDC PHY Write DQ Byte3 Delay Register (MMDC\_MPWRDQBY3DL)

This register is used to add fine-tuning adjustment to every bit in the write DQ byte3 relative to the write DQS

Address: 21B\_0000h base + 838h offset = 21B\_0838h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	wr_dm3_del	wr_dq31_del	0		wr_dq30_del		0		wr_dq29_del		0		wr_dq28_del			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		wr_dq27_del		0		wr_dq26_del		0		wr_dq25_del		0		wr_dq24_del	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### MMDC\_MPWRDQBY3DL field descriptions

Field	Description
31–30 wr_dm3_del	Write dm3 delay fine tuning. This field holds the number of delay units that are added to dm3 relative to dqs3.  00 No change in dm3 delay 01 Add dm3 delay of 1 delay unit. 10 Add dm3 delay of 2 delay units. 11 Add dm3 delay of 3 delay units.
29–28 wr_dq31_del	Write dq31 delay fine tuning. This field holds the number of delay units that are added to dq31 relative to dqs3.  00 No change in dq31 delay 01 Add dq31 delay of 1 delay unit. 10 Add dq31 delay of 2 delay units. 11 Add dq31 delay of 3 delay units.
27–26 Reserved	This read-only field is reserved and always has the value 0.
25–24 wr_dq30_del	Write dq30 delay fine tuning. This field holds the number of delay units that are added to dq30 relative to dqs3.  00 No change in dq30 delay 01 Add dq30 delay of 1 delay unit. 10 Add dq30 delay of 2 delay units. 11 Add dq30 delay of 3 delay units.
23–22 Reserved	This read-only field is reserved and always has the value 0.
21–20 wr_dq29_del	Write dq29 delay fine tuning. This field holds the number of delay units that are added to dq29 relative to dqs3.

Table continues on the next page...

**MMDC\_MPWRDQBY3DL field descriptions (continued)**

Field	Description
	00 No change in dq29 delay 01 Add dq29 delay of 1 delay unit. 10 Add dq29 delay of 2 delay units. 11 Add dq29 delay of 3 delay units.
19–18 Reserved	This read-only field is reserved and always has the value 0.
17–16 wr_dq28_del	Write dq28 delay fine tuning. This field holds the number of delay units that are added to dq28 relative to dqs3.  00 No change in dq28 delay 01 Add dq28 delay of 1 delay unit. 10 Add dq28 delay of 2 delay units. 11 Add dq28 delay of 3 delay units.
15–14 Reserved	This read-only field is reserved and always has the value 0.
13–12 wr_dq27_del	Write dq27 delay fine tuning. This field holds the number of delay units that are added to dq27 relative to dqs3.  00 No change in dq27 delay 01 Add dq27 delay of 1 delay unit. 10 Add dq27 delay of 2 delay units. 11 Add dq27 delay of 3 delay units.
11–10 Reserved	This read-only field is reserved and always has the value 0.
9–8 wr_dq26_del	Write dq26 delay fine tuning. This field holds the number of delay units that are added to dq26 relative to dqs3.  00 No change in dq26 delay 01 Add dq26 delay of 1 delay unit. 10 Add dq26 delay of 2 delay units. 11 Add dq26 delay of 3 delay units.
7–6 Reserved	This read-only field is reserved and always has the value 0.
5–4 wr_dq25_del	Write dq25 delay fine tuning. This field holds the number of delay units that are added to dq25 relative to dqs3.  00 No change in dq25 delay 01 Add dq25 delay of 1 delay unit. 10 Add dq25 delay of 2 delay units. 11 Add dq25 delay of 3 delay units.
3–2 Reserved	This read-only field is reserved and always has the value 0.
1–0 wr_dq24_del	Write dq24 delay fine tuning. This field holds the number of delay units that are added to dq24 relative to dqs3.  00 No change in dq24 delay 01 Add dq24 delay of 1 delay unit.

*Table continues on the next page...*

### MMDC\_MPWRDQBY3DL field descriptions (continued)

Field	Description
10	Add dq24 delay of 2 delay units.
11	Add dq24 delay of 3 delay units.

## 32.12.46 MMDC PHY Read DQS Gating Control Register 0 (MMDC\_MPDGCTRL0)

Address: 21B\_0000h base + 83Ch offset = 21B\_083Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### MMDC\_MPDGCTRL0 field descriptions

Field	Description
31 RST_RD_FIFO	Reset Read Data FIFO and associated pointers. If this bit is asserted then the MMDC resets the read data FIFO and the associated pointers. This bit is self cleared after the FIFO reset is done.
30 DG_CMP_CYC	Read DQS gating sample cycle. If this bit is asserted then the MMDC waits 32 cycles before comparing the read data, Otherwise it waits 16 DDR cycles. 0 MMDC waits 16 DDR cycles 1 MMDC waits 32 DDR cycles
29 DG_DIS	Read DQS gating disable. If this bit is asserted then the MMDC disables the read DQS gating mechnism.

Table continues on the next page...



**MMDC\_MPDGCTRL0 field descriptions (continued)**

Field	Description
	<p>If this bits is asserted (read DQS gating is disabled) then pull-up and pull-down resistors suppose to be used on DQS and DQS# respectively</p> <p>0 Read DQS gating mechanism is enabled 1 Read DQS gating mechanism is disabled</p>
28 HW_DG_EN	<p>Enable automatic read DQS gating calibration. If this bit is asserted then the MMDC performs automatic read DQS gating calibration. HW negates this bit upon completion of the automatic read DQS gating.</p> <p>Note:</p> <p>Before issuing the first read command the MMDC counts 12 cycles.</p> <p>In LPDDR2 mode automatic (HW) read DQS gating should be disabled and Pull-up/pull-down resistors on DQS/DQS# should be enabled while ODT resistors must be disconnected.</p> <p>0 Disable automatic read DQS gating calibration 1 Start automatic read DQS gating calibration</p>
27–24 DG_HC_DEL1	<p>Read DQS gating half cycles delay for Byte1</p> <p>. This field indicates the delay in half cycles between read DQS gate and the middle of the read DQS preamble of Byte1. This delay is added to the delay that is generated by the read DQS1 gating delay-line, So the total read DQS gating delay is <math>(DG\_HC\_DEL\#)*0.5*\text{cycle} + (DG\_DL\_ABS\_OFFSET\#)*1/256*\text{cycle}</math></p> <p>Upon completion of the automatic read DQS gating calibration this field gets the value of the 4 MSB of <math>((HW\_DG\_LOW1 + HW\_DG\_UP1) / 2)</math>.</p> <p>0000 0 cycles delay. 0001 Half cycle delay. 0010 1 cycle delay 1101 6.5 cycles delay 1110 Reserved 1111 Reserved</p>
23 DG_EXT_UP	<p>DG extend upper boundary. By default the upper boundary of DQS gating HW calibration is set according to first failing comparison after at least one passing comparison. If this bit is asserted then the upper boundary is set according to the last passing comparison.</p>
22–16 DG_DL_ABS_OFFSET1	<p>Absolute read DQS gating delay offset for Byte1. This field indicates the absolute delay between read DQS gate and the middle of the read DQS preamble of Byte1 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be <math>(DG\_DL\_ABS\_OFFSET1 / 256)*\text{fast\_clk}</math>.</p> <p>This field can also bit written by HW. Upon completion of the automatic read DQS gating calibration this field gets the value of the 7 LSB of <math>((HW\_DG\_LOW1 + HW\_DG\_UP1) / 2)</math>.</p> <p>Note that not all changes will have effect on the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.</p>
15–13 Reserved	<p>This read-only field is reserved and always has the value 0.</p>
12 HW_DG_ERR	<p>HW DQS gating error. This bit valid is asserted when an error was found during the read DQS gating HW calibration process. Error can occur when no valid value was found during HW calibration.</p> <p>This bit is valid only after HW_DG_EN is de-asserted.</p> <p>0 No error was found during the DQS gating HW calibration process. 1 An error was found during the DQS gating HW calibration process.</p>

*Table continues on the next page...*

### MMDC\_MPDGCTRL0 field descriptions (continued)

Field	Description
11–8 DG_HC_DELO	<p>Read DQS gating half cycles delay for Byte0</p> <p>. This field indicates the delay in half cycles between read DQS gate and the middle of the read DQS preamble of Byte0/4. This delay is added to the delay that is generated by the read DQS1 gating delay-line, So the total read DQS gating delay is <math>(DG\_HC\_DEL\#)*0.5*\text{cycle} + (DG\_DL\_ABS\_OFFSET\#)*1/256*\text{cycle}</math></p> <p>Upon completion of the automatic read DQS gating calibration this field gets the value of the 4 MSB of <math>((HW\_DG\_LOW1 + HW\_DG\_UP1) / 2)</math>.</p> <p>0000 0 cycles delay.  0001 Half cycle delay.  0010 1 cycle delay  1101 6.5 cycles delay  1110 Reserved  1111 Reserved</p>
7 Reserved	This read-only field is reserved and always has the value 0.
6–0 DG_DL_ABS_OFFSET0	<p>Absolute read DQS gating delay offset for Byte0. This field indicates the absolute delay between read DQS gate and the middle of the read DQS preamble of Byte0 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be <math>(DG\_DL\_ABS\_OFFSET0 / 256)*\text{fast\_clk}</math>.</p> <p>This field can also bit written by HW. Upon completion of the automatic read DQS gating calibration this field gets the value of the 7 LSB of <math>((HW\_DG\_LOW0 + HW\_DG\_UP0) / 2)</math>.</p> <p>Note that not all changes will have effect on the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.</p>

## 32.12.47 MMDC PHY Read DQS Gating Control Register 1 (MMDC\_MPDGCTRL1)

Address: 21B\_0000h base + 840h offset = 21B\_0840h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				DG_HC_DEL3				0	DG_DL_ABS_OFFSET3						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				DG_HC_DEL2				0	DG_DL_ABS_OFFSET2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### MMDC\_MPDGCTRL1 field descriptions

Field	Description
31–28 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**MMDC\_MPDGCTRL1 field descriptions (continued)**

Field	Description
27–24 DG_HC_DEL3	<p>Read DQS gating half cycles delay for Byte3</p> <p>. This field indicates the delay in half cycles between read DQS gate and the middle of the read DQS preamble of Byte3/7. This delay is added to the delay that is generated by the read DQS1 gating delay-line, So the total read DQS gating delay is <math>(DG\_HC\_DEL\#)*0.5*\text{cycle} + (DG\_DL\_ABS\_OFFSET\#)*1/256*\text{cycle}</math></p> <p>Upon completion of the automatic read DQS gating calibration this field gets the value of the 4 MSB of <math>((HW\_DG\_LOW3 + HW\_DG\_UP3) / 2)</math>.</p> <p>0000 0 cycles delay.  0001 Half cycle delay.  0010 1 cycle delay  1101 6.5 cycles delay  1110 Reserved  1111 Reserved</p>
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 DG_DL_ABS_OFFSET3	<p>Absolute read DQS gating delay offset for Byte3. This field indicates the absolute delay between read DQS gate and the middle of the read DQS preamble of Byte3 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be <math>(DG\_DL\_ABS\_OFFSET3 / 256)*\text{fast\_clk}</math>.</p> <p>This field can also be written by HW. Upon completion of the automatic read DQS gating calibration this field gets the value of the 7 LSB of <math>((HW\_DG\_LOW3 + HW\_DG\_UP3) / 2)</math>.</p> <p>Note that not all changes will have effect on the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.</p>
15–12 Reserved	This read-only field is reserved and always has the value 0.
11–8 DG_HC_DEL2	<p>Read DQS gating half cycles delay for Byte2</p> <p>. This field indicates the delay in half cycles between read DQS gate and the middle of the read DQS preamble of Byte2/5. This delay is added to the delay that is generated by the read DQS1 gating delay-line, So the total read DQS gating delay is <math>(DG\_HC\_DEL\#)*0.5*\text{cycle} + (DG\_DL\_ABS\_OFFSET\#)*1/256*\text{cycle}</math></p> <p>Upon completion of the automatic read DQS gating calibration this field gets the value of the 4 MSB of <math>((HW\_DG\_LOW2 + HW\_DG\_UP2) / 2)</math>.</p> <p>0000 0 cycles delay.  0001 Half cycle delay.  0010 1 cycle delay  1101 6.5 cycles delay  1110 Reserved  1111 Reserved</p>
7 Reserved	This read-only field is reserved and always has the value 0.
6–0 DG_DL_ABS_OFFSET2	<p>Absolute read DQS gating delay offset for Byte2. This field indicates the absolute delay between read DQS gate and the middle of the read DQS preamble of Byte2 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be <math>(DG\_DL\_ABS\_OFFSET2 / 256)*\text{fast\_clk}</math>.</p> <p>This field can also be written by HW. Upon completion of the automatic read DQS gating calibration this field gets the value of the 7 LSB of <math>((HW\_DG\_LOW2 + HW\_DG\_UP2) / 2)</math>.</p>

*Table continues on the next page...*

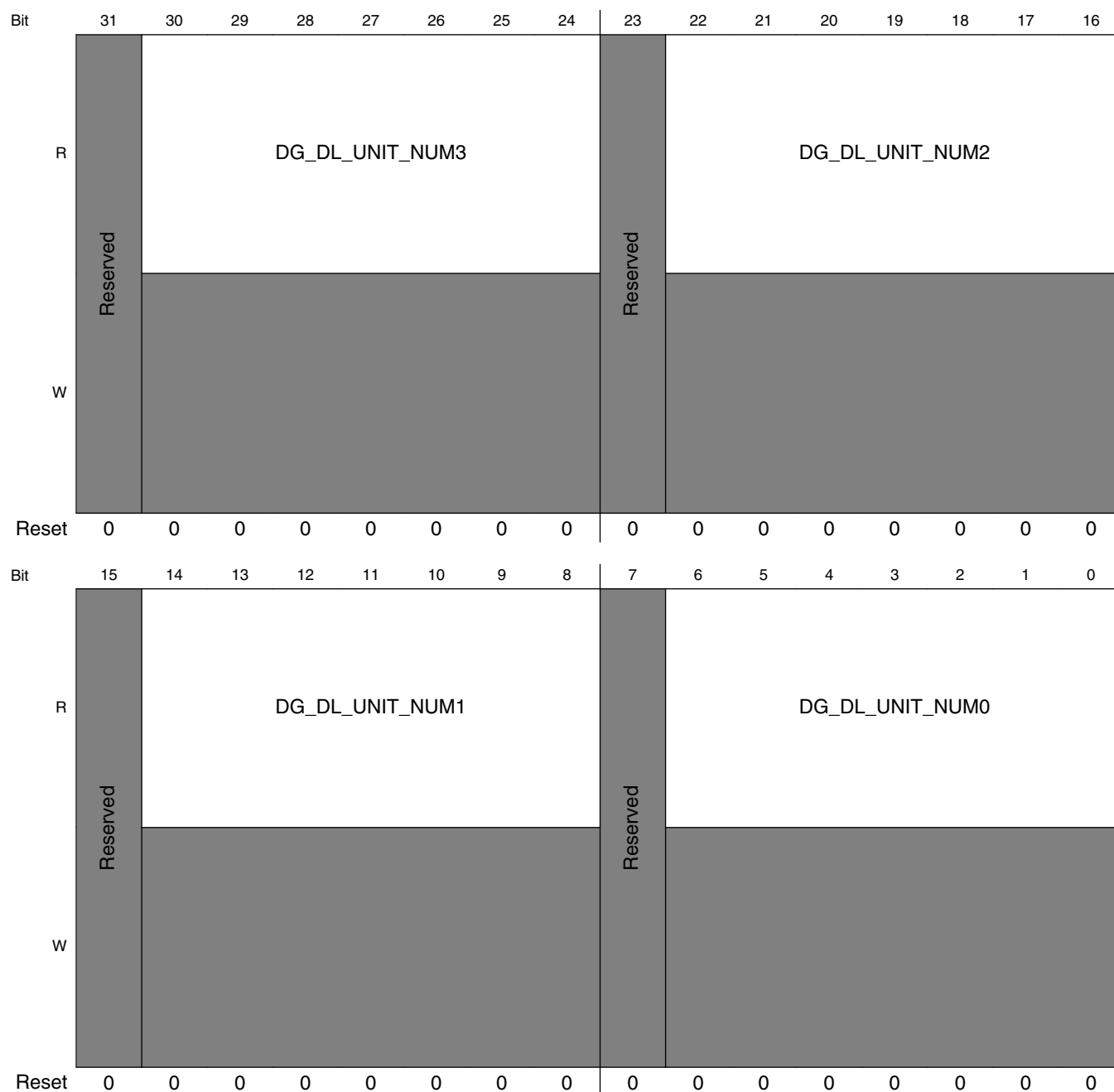
**MMDC\_MPDGCTRL1 field descriptions (continued)**

Field	Description
	Note that not all changes will have effect on the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.

## 32.12.48 MMDC PHY Read DQS Gating delay-line Status Register (MMDC\_MPDGDLST0)

This register holds the status of the 4 dqs gating delay-lines.

Address: 21B\_0000h base + 844h offset = 21B\_0844h



### MMDC\_MPDGDLST0 field descriptions

Field	Description
31 -	This field is reserved. Reserved
30–24 DG_DL_UNIT_NUM3	This field reflects the number of delay units that are actually used by read DQS gating delay-line 3.
23 -	This field is reserved. Reserved
22–16 DG_DL_UNIT_NUM2	This field reflects the number of delay units that are actually used by read DQS gating delay-line 2.
15 -	This field is reserved. Reserved
14–8 DG_DL_UNIT_NUM1	This field reflects the number of delay units that are actually used by read DQS gating delay-line 1.
7 -	This field is reserved. Reserved
6–0 DG_DL_UNIT_NUM0	This field reflects the number of delay units that are actually used by read DQS gating delay-line 0.

## 32.12.49 MMDC PHY Read delay-lines Configuration Register (MMDC\_MPRDDLCTL)

This register controls read delay-lines functionality; it determines DQS delay relative to the associated DQ read access. The delay-line compensates for process variations and produces a constant delay regardless of the process, temperature and voltage.

Address: 21B\_0000h base + 848h offset = 21B\_0848h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	RD_DL_ABS_OFFSET3							0	RD_DL_ABS_OFFSET2						
W																
Reset	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	RD_DL_ABS_OFFSET1							0	RD_DL_ABS_OFFSET0						
W																
Reset	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0

**MMDC\_MPRDDLCTL field descriptions**

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 RD_DL_ABS_OFFSET3	<p>Absolute read delay offset for Byte3. This field indicates the absolute delay between read DQS strobe and the read data of Byte3 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be <math>(RD\_DL\_ABS\_OFFSET3 / 256) * fast\_clk</math>. So for the default value of 64 we get a quarter cycle delay.</p> <p>This field can also bit written by HW. Upon completion of the read delay-line HW calibration this field gets the value of <math>(HW\_RD\_DL\_LOW3 + HW\_RD\_DL\_UP3) / 2</math></p> <p>Note that not all changes will have effect on the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.</p>
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 RD_DL_ABS_OFFSET2	<p>Absolute read delay offset for Byte2. This field indicates the absolute delay between read DQS strobe and the read data of Byte2 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be <math>(RD\_DL\_ABS\_OFFSET2 / 256) * fast\_clk</math>. So for the default value of 64 we get a quarter cycle delay.</p> <p>This field can also bit written by HW. Upon completion of the read delay-line HW calibration this field gets the value of <math>(HW\_RD\_DL\_LOW2 + HW\_RD\_DL\_UP2) / 2</math></p> <p>Note that not all changes will have effect on the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.</p>
15 Reserved	This read-only field is reserved and always has the value 0.
14–8 RD_DL_ABS_OFFSET1	<p>Absolute read delay offset for Byte1. This field indicates the absolute delay between read DQS strobe and the read data of Byte1 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be <math>(RD\_DL\_ABS\_OFFSET1 / 256) * fast\_clk</math>. So for the default value of 64 we get a quarter cycle delay.</p> <p>This field can also bit written by HW. Upon completion of the read delay-line HW calibration this field gets the value of <math>(HW\_RD\_DL\_LOW1 + HW\_RD\_DL\_UP1) / 2</math></p> <p>Note that not all changes will have effect on the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.</p>
7 Reserved	This read-only field is reserved and always has the value 0.
6–0 RD_DL_ABS_OFFSET0	<p>Absolute read delay offset for Byte0. This field indicates the absolute delay between read DQS strobe and the read data of Byte0 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be <math>(RD\_DL\_ABS\_OFFSET0 / 256) * fast\_clk</math>. So for the default value of 64 we get a quarter cycle delay.</p> <p>This field can also bit written by HW. Upon completion of the read delay-line HW calibration this field gets the value of <math>(HW\_RD\_DL\_LOW0 + HW\_RD\_DL\_UP0) / 2</math></p> <p>Note that not all changes will have effect on the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.</p>

## 32.12.50 MMDC PHY Read delay-lines Status Register (MMDC\_MPRDDLST)

This register holds the status of the 4 read delay-lines.

Address: 21B\_0000h base + 84Ch offset = 21B\_084Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	RD_DL_UNIT_NUM3							0	RD_DL_UNIT_NUM2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	RD_DL_UNIT_NUM1							0	RD_DL_UNIT_NUM0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### MMDC\_MPRDDLST field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 RD_DL_UNIT_NUM3	This field reflects the number of delay units that are actually used by read delay-line 3.
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 RD_DL_UNIT_NUM2	This field reflects the number of delay units that are actually used by read delay-line 2.
15 Reserved	This read-only field is reserved and always has the value 0.
14–8 RD_DL_UNIT_NUM1	This field reflects the number of delay units that are actually used by read delay-line 1.
7 Reserved	This read-only field is reserved and always has the value 0.
6–0 RD_DL_UNIT_NUM0	This field reflects the number of delay units that are actually used by read delay-line 0.



## 32.12.51 MMDC PHY Write delay-lines Configuration Register (MMDC\_MPWRDLCTL)

This register controls write delay-lines functionality, it determines DQ/DM delay relative to the associated DQS in write access. The delay-line compensates for process variations, and produces a constant delay regardless of the process, temperature and voltage.

Address: 21B\_0000h base + 850h offset = 21B\_0850h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	WR_DL_ABS_OFFSET3							0	WR_DL_ABS_OFFSET2						
W																
Reset	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	WR_DL_ABS_OFFSET1							0	WR_DL_ABS_OFFSET0						
W																
Reset	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0

### MMDC\_MPWRDLCTL field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 WR_DL_ABS_OFFSET3	<p>Absolute write delay offset for Byte3. This field indicates the absolute delay between write DQS strobe and the write data of Byte3 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be <math>(WR\_DL\_ABS\_OFFSET3 / 256) * fast\_clk</math>. So for the default value of 64 we get a quarter cycle delay.</p> <p>This field can also bit written by HW. Upon completion of the write delay-line HW calibration this field gets the value of <math>(HW\_WR\_DL\_LOW3 + HW\_WR\_DL\_UP3) / 2</math></p> <p>Note that not all changes will have effect on the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.</p>
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 WR_DL_ABS_OFFSET2	<p>Absolute write delay offset for Byte2. This field indicates the absolute delay between write DQS strobe and the write data of Byte2 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be <math>(WR\_DL\_ABS\_OFFSET2 / 256) * fast\_clk</math>. So for the default value of 64 we get a quarter cycle delay.</p> <p>This field can also bit written by HW. Upon completion of the write delay-line HW calibration this field gets the value of <math>(HW\_WR\_DL\_LOW2 + HW\_WR\_DL\_UP2) / 2</math></p> <p>Note that not all changes will have effect on the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.</p>
15 Reserved	This read-only field is reserved and always has the value 0.
14–8 WR_DL_ABS_OFFSET1	<p>Absolute write delay offset for Byte1. This field indicates the absolute delay between write DQS strobe and the write data of Byte1 with fractions of a clock period and up to half cycle. The fraction is process and</p>

Table continues on the next page...

### MMDC\_MPWRDLCTL field descriptions (continued)

Field	Description
	frequency independent. The delay of the delay-line would be $(WR\_DL\_ABS\_OFFSET1 / 256) * fast\_clk$ . So for the default value of 64 we get a quarter cycle delay.  This field can also bit written by HW. Upon completion of the write delay-line HW calibration this field gets the value of $(HW\_WR\_DL\_LOW1 + HW\_WR\_DL\_UP1) / 2$  Note that not all changes of this value will affect the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.
7 Reserved	This read-only field is reserved and always has the value 0.
6–0 WR_DL_ABS_OFFSET0	Absolute write delay offset for Byte0. This field indicates the absolute delay between write DQS strobe and the write data of Byte3 with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be $(WR\_DL\_ABS\_OFFSET0 / 256) * fast\_clk$ . So for the default value of 64 we get a quarter cycle delay.  This field can also bit written by HW. Upon completion of the write delay-line HW calibration this field gets the value of $(HW\_WR\_DL\_LOW0 + HW\_WR\_DL\_UP0) / 2$  Note that not all changes of this value will affect the actual delay. If the requested change is smaller than the delay-line resolution, then no change will occur.

### 32.12.52 MMDC PHY Write delay-lines Status Register (MMDC\_MPWRDLST)

This register holds the status of the 4 write delay-line.

Address: 21B\_0000h base + 854h offset = 21B\_0854h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	WR_DL_UNIT_NUM3							0	WR_DL_UNIT_NUM2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	WR_DL_UNIT_NUM1							0	WR_DL_UNIT_NUM0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### MMDC\_MPWRDLST field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 WR_DL_UNIT_NUM3	This field reflects the number of delay units that are actually used by write delay-line 3.

Table continues on the next page...

**MMDC\_MPWRDLST field descriptions (continued)**

Field	Description
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 WR_DL_UNIT_NUM2	This field reflects the number of delay units that are actually used by write delay-line 2.
15 Reserved	This read-only field is reserved and always has the value 0.
14–8 WR_DL_UNIT_NUM1	This field reflects the number of delay units that are actually used by write delay-line 1.
7 Reserved	This read-only field is reserved and always has the value 0.
6–0 WR_DL_UNIT_NUM0	This field reflects the number of delay units that are actually used by write delay-line 0.

**32.12.53 MMDC PHY CK Control Register (MMDC\_MPSDCTRL)**

This register controls the fine tuning of the primary clock (CK0).

Address: 21B\_0000h base + 858h offset = 21B\_0858h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				SDCLK1_del				0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MPSDCTRL field descriptions**

Field	Description
31–12 Reserved	This read-only field is reserved and always has the value 0.
11–10 SDCLK1_del	DDR clock1 delay fine tuning. This field holds the number of delay units that are added to DDR clock1 (CK1).  00 No change in DDR clock delay 01 Add DDR clock delay of 1 delay unit. 10 Add DDR clock delay of 2 delay units. 11 Add DDR clock delay of 3 delay units.

*Table continues on the next page...*

**MMDC\_MPSPDCTRL field descriptions (continued)**

Field	Description
9–8 SDclk0_del	DDR clock0 delay fine tuning. This field holds the number of delay units that are added to DDR clock (CK0).  00 No change in DDR clock0 delay 01 Add DDR clock0 delay of 1 delay unit. 10 Add DDR clock0 delay of 2 delay units. 11 Add DDR clock0 delay of 3 delay units.
7–0 Reserved	This read-only field is reserved and always has the value 0.

**32.12.54 MMDC ZQ LPDDR2 HW Control Register (MMDC\_MPZQLP2CTL)**

This register controls the idle time that takes the LPDDR2 device to perform ZQ calibration

Address: 21B\_0000h base + 85Ch offset = 21B\_085Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	0	ZQ_LP2_HW_ZQCS								ZQ_LP2_HW_ZQCL							
W																	
Reset	0	0	0	1	1	0	1	1	0	1	0	1	1	1	1	1	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	0								ZQ_LP2_HW_ZQINIT								
W																	
Reset	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	1	

**MMDC\_MPZQLP2CTL field descriptions**

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 ZQ_LP2_HW_ZQCS	This register defines the period in cycles that it takes the memory device to perform a Short ZQ calibration. This is the period of time that the MMDC has to wait after sending a long ZQ calibration and before sending other commands. This delay will also be used if ZQ reset is sent.  0x0–0x1A Reserved 0x1B 112 cycles (default) 0x1C 116 cycles 0x7E 508 cycles 0x7F 512 cycles

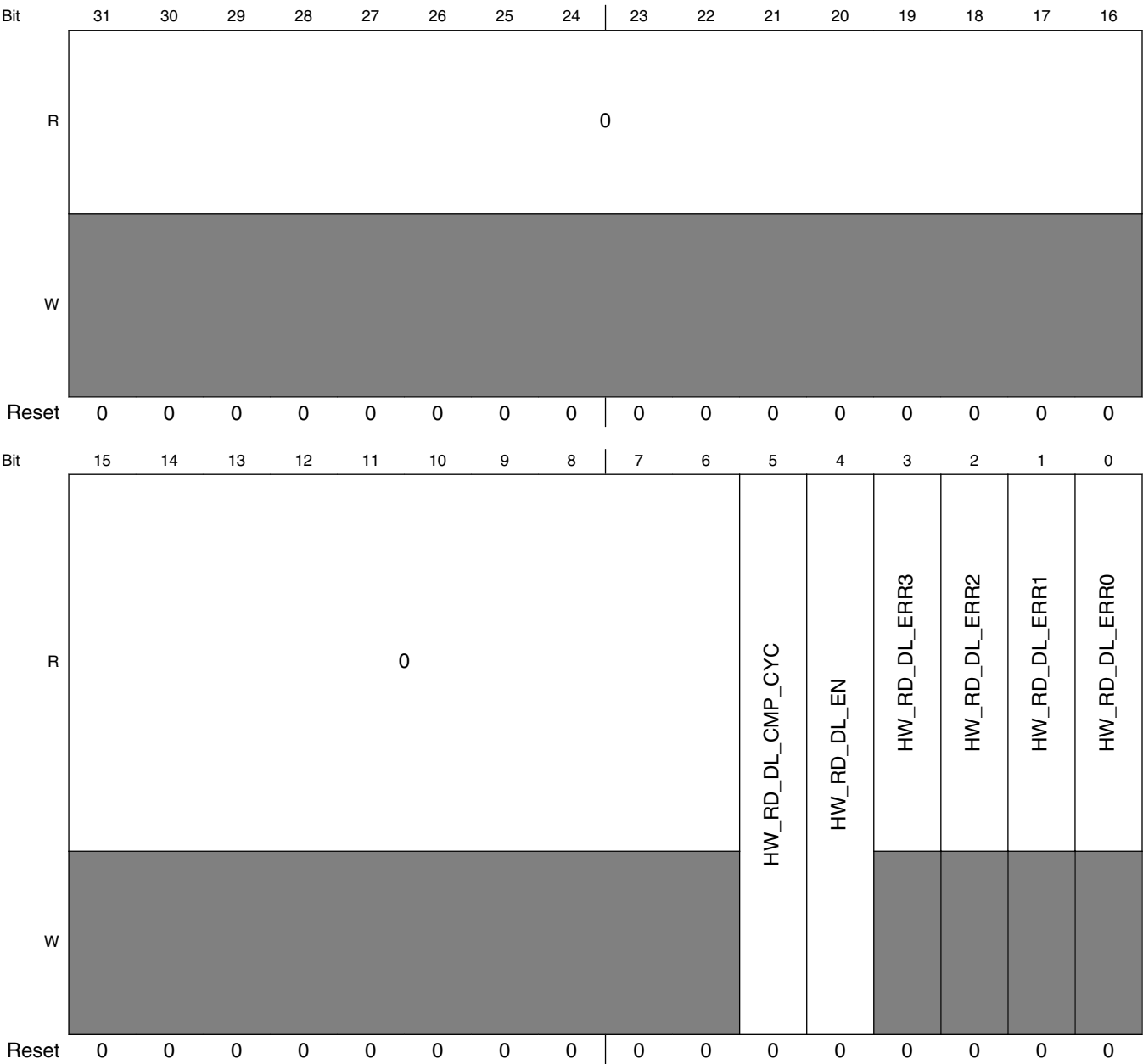
Table continues on the next page...

**MMDC\_MPZQLP2CTL field descriptions (continued)**

Field	Description
23–16 ZQ_LP2_HW_ ZQCL	<p>This register defines the period in cycles that it takes the memory device to perform a long ZQ calibration. This is the period of time that the MMDC has to wait after sending a Short ZQ calibration and before sending other commands.</p> <p>0x0-0x36    Reserved</p> <p>0x37        112 cycles</p> <p>0x38        114 cycles</p> <p>0x5F        192 cycles (Default, JEDEC value, tZQCL, for LPDDR2, 360ns @ clock frequency 533MHz)</p> <p>0xFE        510 cycles</p> <p>0xFF        512 cycles</p>
15–9 Reserved	This read-only field is reserved and always has the value 0.
8–0 ZQ_LP2_HW_ ZQINIT	<p>This register defines the period in cycles that it takes the memory device to perform a Init ZQ calibration. This is the period of time that the MMDC has to wait after sending a init ZQ calibration and before sending other commands.</p> <p>0x0-0x36    Reserved</p> <p>0x37        112 cycles</p> <p>0x38        114 cycles</p> <p>0x109       532 cycles (Default, JEDEC value, tZQINIT, for LPDDR2, 1us @ clock frequency 533MHz)</p> <p>0x1FE       1022 cycles</p> <p>0x1FF       1024 cycles</p>

32.12.55 MMDC PHY Read Delay HW Calibration Control Register (MMDC\_MPRDDLHWCTL)

Address: 21B\_0000h base + 860h offset = 21B\_0860h



MMDC\_MPRDDLHWCTL field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value 0.

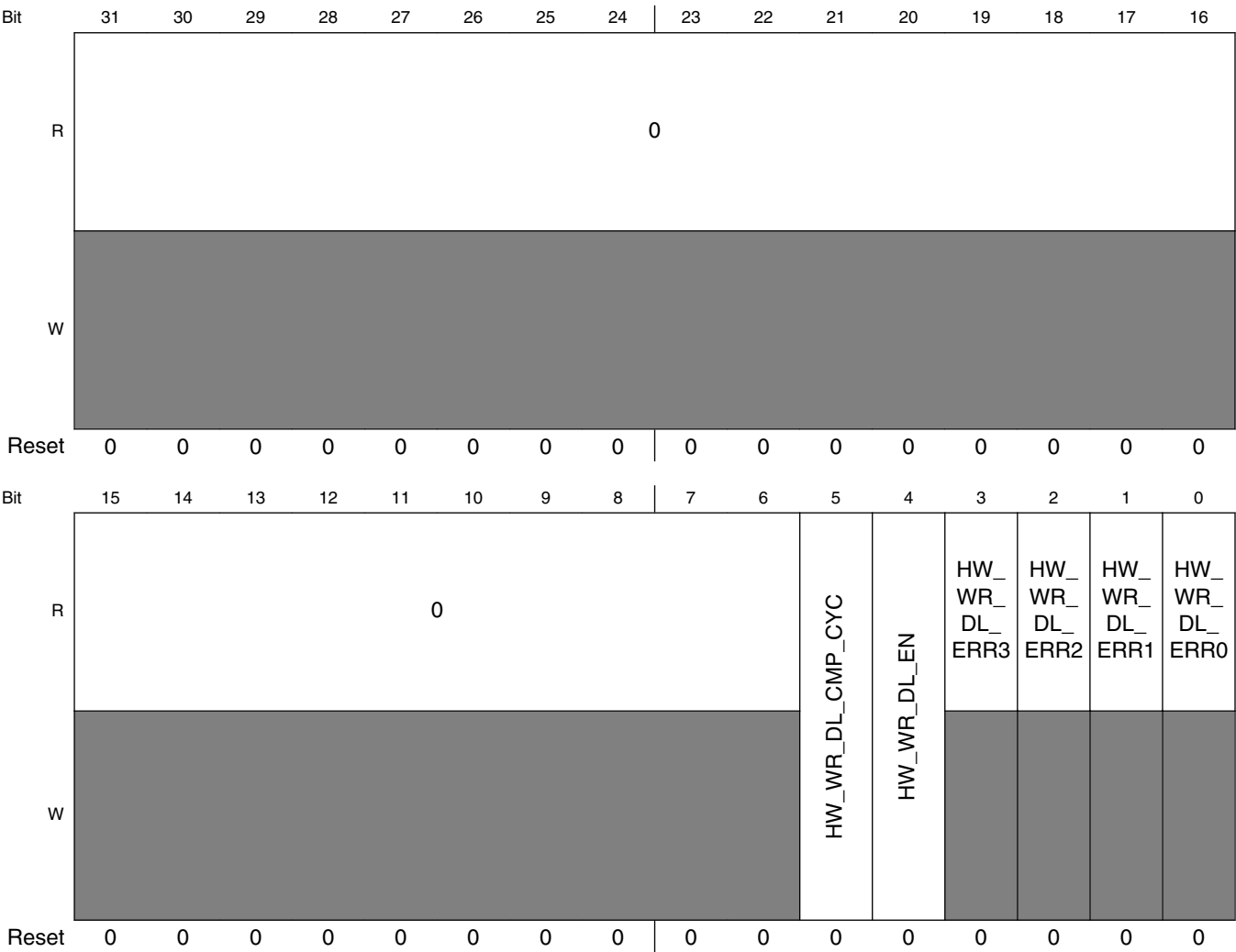
Table continues on the next page...

**MMDC\_MPRDDLHWCTL field descriptions (continued)**

<b>Field</b>	<b>Description</b>
5 HW_RD_DL_CMP_CYC	Automatic (HW) read sample cycle. If this bit is asserted then the MMDC will compare the read data 32 cycles after the MMDC sent the read command enable pulse else it compares the data after 16 cycles.
4 HW_RD_DL_EN	Enable automatic (HW) read calibration. If this bit is asserted then the MMDC will perform an automatic read calibration. HW should negate this bit upon completion of the calibration. Negation of this bit also points that the read calibration results are valid  Note: Before issuing the first read command MMDC counts 12 cycles.
3 HW_RD_DL_ERR3	Automatic (HW) read calibration error of Byte3. If this bit is asserted then it indicates that an error was found during the HW calibration process of read delay-line 3. In case this bit is zero at the end of the calibration process then the boundary results can be found at MPRDDLHWST1 register. This bit is valid only after HW_RD_DL_EN is de-asserted.  0 No error was found in read delay-line 3 during the automatic (HW) read calibration process of read delay-line 3. 1 An error was found in read delay-line 3 during the automatic (HW) read calibration process of read delay-line 3.
2 HW_RD_DL_ERR2	Automatic (HW) read calibration error of Byte2. If this bit is asserted then it indicates that an error was found during the HW calibration process of read delay-line 2. In case this bit is zero at the end of the calibration process then the boundary results can be found at MPRDDLHWST1 register. This bit is valid only after HW_RD_DL_EN is de-asserted.  0 No error was found in read delay-line 2 during the automatic (HW) read calibration process of read delay-line 2. 1 An error was found in read delay-line 2 during the automatic (HW) read calibration process of read delay-line 2.
1 HW_RD_DL_ERR1	Automatic (HW) read calibration error of Byte1. If this bit is asserted then it indicates that an error was found during the HW calibration process of read delay-line 1. In case this bit is zero at the end of the calibration process then the boundary results can be found at MPRDDLHWST0 register. This bit is valid only after HW_RD_DL_EN is de-asserted.  0 No error was found in read delay-line 1 during the automatic (HW) read calibration process of read delay-line 1. 1 An error was found in read delay-line 1 during the automatic (HW) read calibration process of read delay-line 1.
0 HW_RD_DL_ERR0	Automatic (HW) read calibration error of Byte0. If this bit is asserted then it indicates that an error was found during the HW calibration process of read delay-line 0. In case this bit is zero at the end of the calibration process then the boundary results can be found at MPRDDLHWST0 register. This bit is valid only after HW_RD_DL_EN is de-asserted.  0 No error was found in read delay-line 0 during the automatic (HW) read calibration process of read delay-line 0. 1 An error was found in read delay-line 0 during the automatic (HW) read calibration process of read delay-line 0.

32.12.56 MMDC PHY Write Delay HW Calibration Control Register (MMDC\_MPWRDLHWCTL)

Address: 21B\_0000h base + 864h offset = 21B\_0864h



MMDC\_MPWRDLHWCTL field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value 0.
5 HW_WR_DL_CMP_CYC	Write sample cycle. If this bit is asserted then the MMDC will compare the data 32 cycles after the MMDC sent the read command enable pulse else it compares the data after 16 cycles.
4 HW_WR_DL_EN	Enable automatic (HW) write calibration. If this bit is asserted then the MMDC will perform an automatic write calibration. HW should negate this bit upon completion of the calibration. Negation of this bit also indicates that the write calibration results are valid  Note: Before issuing the first read command MMDC counts 12 cycles.

Table continues on the next page...



**MMDC\_MPWRDLHWCTL field descriptions (continued)**

Field	Description
3 HW_WR_DL_ERR3	Automatic (HW) write calibration error of Byte3. If this bit is asserted then it indicates that an error was found during the HW calibration process of write delay-line 3. In case this bit is zero at the end of the calibration process then the boundary results can be found at MPWRDLHWST1 register. This bit is valid only after HW_WR_DL_EN is de-asserted.  0 No error was found during the automatic (HW) write calibration process of write delay-line 3. 1 An error was found during the automatic (HW) write calibration process of write delay-line 3.
2 HW_WR_DL_ERR2	Automatic (HW) write calibration error of Byte2. If this bit is asserted then it indicates that an error was found during the HW calibration process of write delay-line 2. In case this bit is zero at the end of the calibration process then the boundary results can be found at MPWRDLHWST1 register. This bit is valid only after HW_WR_DL_EN is de-asserted.  0 No error was found during the automatic (HW) write calibration process of write delay-line 2. 1 An error was found during the automatic (HW) write calibration process of write delay-line 2.
1 HW_WR_DL_ERR1	Automatic (HW) write calibration error of Byte1. If this bit is asserted then it indicates that an error was found during the HW calibration process of write delay-line 1. In case this bit is zero at the end of the calibration process then the boundary results can be found at MPWRDLHWST0 register. This bit is valid only after HW_WR_DL_EN is de-asserted.  0 No error was found during the automatic (HW) write calibration process of write delay-line 1. 1 An error was found during the automatic (HW) write calibration process of write delay-line 1.
0 HW_WR_DL_ERR0	Automatic (HW) write calibration error of Byte0. If this bit is asserted then it indicates that an error was found during the HW calibration process of write delay-line 0. In case this bit is zero at the end of the calibration process then the boundary results can be found at MPWRDLHWST0 register. This bit is valid only after HW_WR_DL_EN is de-asserted.  0 No error was found during the automatic (HW) write calibration process of write delay-line 0. 1 An error was found during the automatic (HW) write calibration process of write delay-line 0.

**32.12.57 MMDC PHY Read Delay HW Calibration Status Register 0 (MMDC\_MPRDDLHWST0)**

Address: 21B\_0000h base + 868h offset = 21B\_0868h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	HW_RD_DL_UP1							0	HW_RD_DL_LOW1						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	HW_RD_DL_UP0							0	HW_RD_DL_LOW0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MPRDDLHWST0 field descriptions**

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 HW_RD_DL_UP1	Automatic (HW) read calibration result of the upper boundary of Byte1. This field holds the automatic (HW) read calibration result of the upper boundary of Byte1
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 HW_RD_DL_LOW1	Automatic (HW) read calibration result of the lower boundary of Byte1. This field holds the automatic (HW) read calibration result of the lower boundary of Byte1
15 Reserved	This read-only field is reserved and always has the value 0.
14–8 HW_RD_DL_UP0	Automatic (HW) read calibration result of the upper boundary of Byte0. This field holds the automatic (HW) read calibration result of the upper boundary of Byte0.
7 Reserved	This read-only field is reserved and always has the value 0.
6–0 HW_RD_DL_LOW0	Automatic (HW) read calibration result of the lower boundary of Byte0. This field holds the automatic (HW) read calibration result of the lower boundary of Byte0.

**32.12.58 MMDC PHY Read Delay HW Calibration Status Register 1 (MMDC\_MPRDDLHWST1)**

Address: 21B\_0000h base + 86Ch offset = 21B\_086Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	HW_RD_DL_UP3							0	HW_RD_DL_LOW3						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	HW_RD_DL_UP2							0	HW_RD_DL_LOW2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MPRDDLHWST1 field descriptions**

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 HW_RD_DL_UP3	Automatic (HW) read calibration result of the upper boundary of Byte3. This field holds the automatic (HW) read calibration result of the upper boundary of Byte3

*Table continues on the next page...*

**MMDC\_MPRDDLHWST1 field descriptions (continued)**

Field	Description
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 HW_RD_DL_LOW3	Automatic (HW) read calibration result of the lower boundary of Byte3. This field holds the automatic (HW) read calibration result of the lower boundary of Byte3
15 Reserved	This read-only field is reserved and always has the value 0.
14–8 HW_RD_DL_UP2	Automatic (HW) read calibration result of the upper boundary of Byte2. This field holds the automatic (HW) read calibration result of the upper boundary of Byte2.
7 Reserved	This read-only field is reserved and always has the value 0.
6–0 HW_RD_DL_LOW2	Automatic (HW) read calibration result of the lower boundary of Byte2. This field holds the automatic (HW) read calibration result of the lower boundary of Byte2.

**32.12.59 MMDC PHY Write Delay HW Calibration Status Register 0 (MMDC\_MPWRDLHWST0)**

Address: 21B\_0000h base + 870h offset = 21B\_0870h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	HW_WR_DL_UP1							0	HW_WR_DL_LOW1						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	HW_WR_DL_UP0							0	HW_WR_DL_LOW0						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MPWRDLHWST0 field descriptions**

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 HW_WR_DL_UP1	Automatic (HW) write calibration result of the upper boundary of Byte1. This field holds the automatic (HW) write calibration result of the upper boundary of Byte1.
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 HW_WR_DL_LOW1	Automatic (HW) write calibration result of the lower boundary of Byte1. This field holds the automatic (HW) write calibration result of the lower boundary of Byte1.

*Table continues on the next page...*

### MMDC\_MPWRDLHWST0 field descriptions (continued)

Field	Description
15 Reserved	This read-only field is reserved and always has the value 0.
14–8 HW_WR_DL_UP0	Automatic (HW) write calibration result of the upper boundary of Byte0. This field holds the automatic (HW) write calibration result of the upper boundary of Byte0.
7 Reserved	This read-only field is reserved and always has the value 0.
6–0 HW_WR_DL_LOW0	Automatic (HW) write calibration result of the lower boundary of Byte0. This field holds the automatic (HW) write calibration result of the lower boundary of Byte0.

## 32.12.60 MMDC PHY Write Delay HW Calibration Status Register 1 (MMDC\_MPWRDLHWST1)

Address: 21B\_0000h base + 874h offset = 21B\_0874h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	HW_WR_DL_UP3							0	HW_WR_DL_LOW3						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	HW_WR_DL_UP2							0	HW_WR_DL_LOW2						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### MMDC\_MPWRDLHWST1 field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 HW_WR_DL_UP3	Automatic (HW) write calibration result of the upper boundary of Byte3. This field holds the automatic (HW) write calibration result of the upper boundary of Byte3.
23 Reserved	This read-only field is reserved and always has the value 0.
22–16 HW_WR_DL_LOW3	Automatic (HW) write calibration result of the lower boundary of Byte3. This field holds the automatic (HW) write calibration result of the lower boundary of Byte3.
15 Reserved	This read-only field is reserved and always has the value 0.
14–8 HW_WR_DL_UP2	Automatic (HW) write calibration result of the upper boundary of Byte2. This field holds the automatic (HW) write calibration result of the upper boundary of Byte2.

Table continues on the next page...

**MMDC\_MPWRDLHWST1 field descriptions (continued)**

Field	Description
7 Reserved	This read-only field is reserved and always has the value 0.
6–0 HW_WR_DL_LOW2	Automatic (HW) write calibration result of the lower boundary of Byte2. This field holds the automatic (HW) write calibration result of the lower boundary of Byte2.

**32.12.61 MMDC PHY Write Leveling HW Error Register (MMDC\_MPWLHWERR)**

Address: 21B\_0000h base + 878h offset = 21B\_0878h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HW_WL3_DQ								HW_WL2_DQ								HW_WL1_DQ								HW_WL0_DQ							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MPWLHWERR field descriptions**

Field	Description
31–24 HW_WL3_DQ	HW write-leveling calibration result of Byte3. This field holds the results for all the 8 write-leveling steps of Byte3. i.e bit 0 holds the result of the write-leveling calibration of 0 delay, bit 1 holds the result of the write-leveling calibration of 1/8delay till bit 7 that holds the result of the write-leveling calibration of 7/8 delay
23–16 HW_WL2_DQ	HW write-leveling calibration result of Byte2. This field holds the results for all the 8 write-leveling steps of Byte2. i.e bit 0 holds the result of the write-leveling calibration of 0 delay, bit 1 holds the result of the write-leveling calibration of 1/8delay till bit 7 that holds the result of the write-leveling calibration of 7/8 delay
15–8 HW_WL1_DQ	HW write-leveling calibration result of Byte1. This field holds the results for all the 8 write-leveling steps of Byte1. i.e bit 0 holds the result of the write-leveling calibration of 0 delay, bit 1 holds the result of the write-leveling calibration of 1/8delay till bit 7 that holds the result of the write-leveling calibration of 7/8 delay
7–0 HW_WL0_DQ	HW write-leveling calibration result of Byte0. This field holds the results for all the 8 write-leveling steps of Byte0. i.e bit 0 holds the result of the write-leveling calibration of 0 delay, bit 1 holds the result of the write-leveling calibration of 1/8delay till bit 7 that holds the result of the write-leveling calibration of 7/8 delay

**32.12.62 MMDC PHY Read DQS Gating HW Status Register 0 (MMDC\_MPDGHWST0)**

Address: 21B\_0000h base + 87Ch offset = 21B\_087Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								HW_DG_UP0								Reserved								HW_DG_LOW0							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### MMDC\_MPDGHWST0 field descriptions

Field	Description
31–27 -	This field is reserved. Reserved
26–16 HW_DG_UP0	HW DQS gating calibration result of the upper boundary of Byte0. This field holds the HW DQS gating calibration result of the upper boundary of Byte0.
15–11 -	This field is reserved. Reserved
10–0 HW_DG_LOW0	HW DQS gating calibration result of the lower boundary of Byte0. This field holds the HW DQS gating calibration result of the lower boundary of Byte0.

## 32.12.63 MMDC PHY Read DQS Gating HW Status Register 1 (MMDC\_MPDGHWST1)

Address: 21B\_0000h base + 880h offset = 21B\_0880h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					HW_DG_UP1											Reserved					HW_DG_LOW1										
W	Reserved					Reserved											Reserved					Reserved										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### MMDC\_MPDGHWST1 field descriptions

Field	Description
31–27 -	This field is reserved. Reserved
26–16 HW_DG_UP1	HW DQS gating calibration result of the upper boundary of Byte1. This field holds the HW DQS gating calibration result of the upper boundary of Byte1.
15–11 -	This field is reserved. Reserved
10–0 HW_DG_LOW1	HW DQS gating calibration result of the lower boundary of Byte1. This field holds the HW DQS gating calibration result of the lower boundary of Byte1.

## 32.12.64 MMDC PHY Read DQS Gating HW Status Register 2 (MMDC\_MPDGHWST2)

Address: 21B\_0000h base + 884h offset = 21B\_0884h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					HW_DG_UP2											Reserved					HW_DG_LOW2										
W	Reserved					Reserved											Reserved					Reserved										
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MPDGHWST2 field descriptions**

Field	Description
31–27 -	This field is reserved. Reserved
26–16 HW_DG_UP2	HW DQS gating calibration result of the upper boundary of Byte2. This field holds the HW DQS gating calibration result of the upper boundary of Byte2.
15–11 -	This field is reserved. Reserved
10–0 HW_DG_LOW2	HW DQS gating calibration result of the lower boundary of Byte2. This field holds the HW DQS gating calibration result of the lower boundary of Byte2.

**32.12.65 MMDC PHY Read DQS Gating HW Status Register 3 (MMDC\_MPDGHWST3)**

Address: 21B\_0000h base + 888h offset = 21B\_0888h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					HW_DG_UP3											Reserved					HW_DG_LOW3										
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**MMDC\_MPDGHWST3 field descriptions**

Field	Description
31–27 -	This field is reserved. Reserved
26–16 HW_DG_UP3	HW DQS gating calibration result of the upper boundary of Byte3. This field holds the HW DQS gating calibration result of the upper boundary of Byte3.
15–11 -	This field is reserved. Reserved
10–0 HW_DG_LOW3	HW DQS gating calibration result of the lower boundary of Byte3. This field holds the HW DQS gating calibration result of the lower boundary of Byte3.

### 32.12.66 MMDC PHY Pre-defined Compare Register 1 (MMDC\_MPPDCMPR1)

This register holds the MMDC pre-defined compare value that will be used during automatic read, read DQS gating and write calibration process. The compare value can be the MPR value (as defined in the JEDEC) or can be programmed by the PDV1 and PDV2 fields. In case of DDR3 (BL=8) the MMDC will duplicate PDV1,PDV2 and drive that data on Beat4-7 of the same byte

Address: 21B\_0000h base + 88Ch offset = 21B\_088Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PDV2																PDV1															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### MMDC\_MPPDCMPR1 field descriptions

Field	Description
31–16 PDV2	MMDC Pre defined compare value2. This field holds the 2 MSB of the data that will be driven to the DDR device during automatic read, read DQS gating and write calibrations in case MPR(DDR3)/ DQ calibration (LPDDR2) mode are disabled (MPR_CMP is disabled). Upon read access during the calibration the MMDC will compare the read data with the data that is stored in this field.  Note : Before issue the read access the MMDC will invert the value of this field and drive it to the associate entry in the read comparison FIFO. For further information see Section 19.14.3.1.2, "Calibration with pre-defined value", Section 19.14.4.1.2, "Calibration with pre-defined value and Section 19.14.5.1, "HW (automatic) Write Calibraion
15–0 PDV1	MMDC Pre defined compare value2. This field holds the 2 LSB of the data that will be driven to the DDR device during automatic read, read DQS gating and write calibrations in case MPR(DDR3)/ DQ calibration (LPDDR2) mode are disabled (MPR_CMP is disabled). Upon read access during the calibration the MMDC will compare the read data with the data that is stored in this field.  <b>NOTE:</b> Before issuing the read access, the MMDC will invert the value of this field and drive it to the associated entry in the read comparison FIFO.



## 32.12.67 MMDC PHY Pre-defined Compare and CA delay-line Configuration Register (MMDC\_MPPDCMPR2)

Address: 21B\_0000h base + 890h offset = 21B\_0890h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16		
R	0	PHY_CA_DL_UNIT												0				
W															CA_DL_ABS_OFFSET			
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0		

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
R	0														READ_LEVEL_PATTERN		MPR_FULL_CMP	MPR_CMP
W																		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

**MMDC\_MPPDCMPR2 field descriptions**

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–24 PHY_CA_DL_UNIT	This field reflects the number of delay units that are actually used by CA (Command/Address of LPDDR2) delay-line
23 Reserved	This read-only field is reserved and always has the value 0.

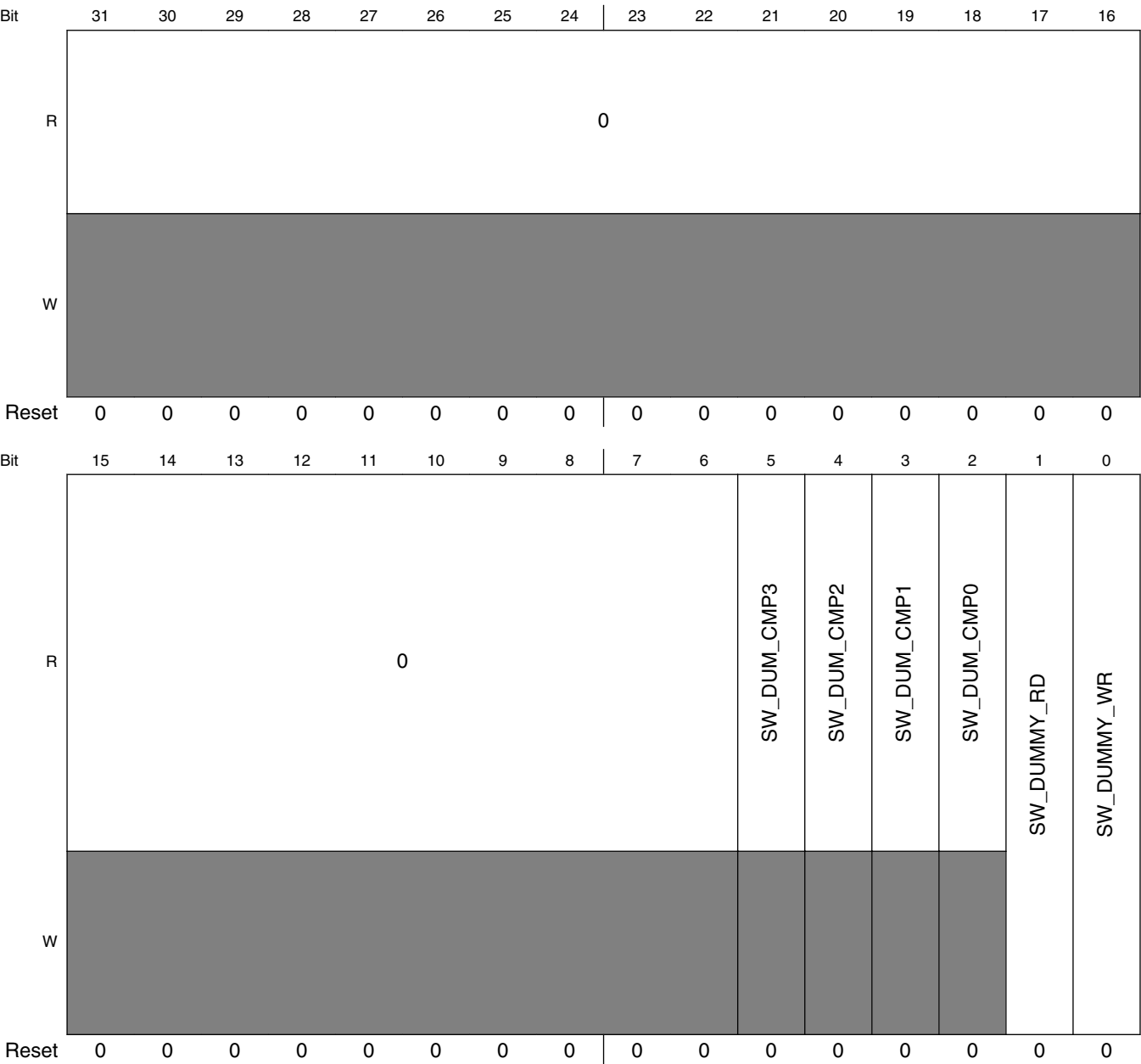
*Table continues on the next page...*

### MMDC\_MPPDCMPR2 field descriptions (continued)

Field	Description
22–16 CA_DL_ABS_OFFSET	Absolute CA (Command/Address of LPDDR2) offset. This field indicates the absolute delay between CA (Command/Address) bus and the DDR clock (CK) with fractions of a clock period and up to half cycle. The fraction is process and frequency independent. The delay of the delay-line would be $(CA\_DL\_ABS\_OFFSET / 256) * fast\_clk$ . So for the default value of 64 we get a quarter cycle delay.
15–3 Reserved	This read-only field is reserved and always has the value 0.
2 READ_LEVEL_PATTERN	MPR(DDR3)/DQ calibration(LPDDR2) read compare pattern. In case MPR(DDR3)/DQ calibration(LPDDR2) modes are used during the calibration process (MPR_CMP is asserted) then this field indicates the read pattern for the comparison.  0 Compare with read pattern 1010 1 Compare with read pattern 0011 (Used only in LPDDR2 mode)
1 MPR_FULL_CMP	MPR(DDR3)/DQ calibration (LPDDR2) full compare enable. In case MPR(DDR3)/DQ calibration(LPDDR2) modes are used during the calibration process (MPR_CMP is asserted) then this field indicates whether the MMDC will compare all the bits of the data that is read from the DDR device to the MPR pre-defined pattern. When this bit is de-asserted only LSB of each byte is compared.
0 MPR_CMP	MPR(DDR3)/DQ calibration (LPDDR2) compare enable. This bit indicates whether the MMDC will compare the read data during automatic read and read DQS calibration processes to the pre-defined patterns that are driven by the DDR device (READ_LEVEL_PATTERN as defined by JEDEC) or general pre-defined value that are stored in PDV1 and PDV2. When this bit is disabled data is compared to the data of the pre defined compare value field  For further information see <a href="#">Read DQS Gating Calibration</a> and <a href="#">Read Calibration</a> .

### 32.12.68 MMDC PHY SW Dummy Access Register (MMDC\_MPSWDAR0)

Address: 21B\_0000h base + 894h offset = 21B\_0894h



MMDC\_MPSWDAR0 field descriptions

Field	Description
31–6 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...

**MMDC\_MPSWDAR0 field descriptions (continued)**

Field	Description
5 SW_DUM_CMP3	SW dummy read byte3 compare results. This bit indicates the result of the read data comparison of Byte3 at the completion of SW_DUMMY_RD. This bit is valid only when SW_DUMMY_RD is de-asserted.  0 Dummy read fail 1 Dummy read pass
4 SW_DUM_CMP2	SW dummy read byte2 compare results. This bit indicates the result of the read data comparison of Byte2 at the completion of SW_DUMMY_RD. This bit is valid only when SW_DUMMY_RD is de-asserted.  0 Dummy read fail 1 Dummy read pass
3 SW_DUM_CMP1	SW dummy read byte1 compare results. This bit indicates the result of the read data comparison of Byte1 at the completion of SW_DUMMY_RD. This bit is valid only when SW_DUMMY_RD is de-asserted.  0 Dummy read fail 1 Dummy read pass
2 SW_DUM_CMP0	SW dummy read byte0 compare results. This bit indicates the result of the read data comparison of Byte0 at the completion of SW_DUMMY_RD. This bit is valid only when SW_DUMMY_RD is de-asserted.  0 Dummy read fail 1 Dummy read pass
1 SW_DUMMY_RD	SW dummy read. When this bit is asserted the MMDC will generate internally read access without intervention of the system toward bank 0, row 0, column 0. If MPR_CMP = 1 then the read data will be compared to MPPDCMPR2[READ_LEVEL_PATTERN]. If MPR_CMP = 0 then the read data will be compared to MPPDCMPR1[PDV1], MPPDCMPR1[PDV2]. Upon completion of the access this bit is de-asserted automatically and the read data and comparison results are valid at MPSWDAR0[SW_DUM_CMP#] and MPSWDRDR0-MPSWDRDR7 respectively.
0 SW_DUMMY_WR	SW dummy write. When this bit is asserted the MMDC will generate internally write access without intervention of the system toward bank 0, row 0, column 0, while the data is driven from MPPDCMPR1[PDV1] and MPPDCMPR1[PDV2]. The bit is de-asserted automatically upon completion of the access.

**32.12.69 MMDC PHY SW Dummy Read Data Register 0 (MMDC\_MPSWDRDR0)**

Address: 21B\_0000h base + 898h offset = 21B\_0898h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DUM_RD0																															
W																																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

**MMDC\_MPSWDRDR0 field descriptions**

Field	Description
31–0 DUM_RD0	Dummy read data0. This field holds the first data that is read from the DDR during SW dummy read access (i.e when SW_DUMMY_RD = 1). This field is valid only when SW_DUMMY_RD is de-asserted

### 32.12.70 MMDC PHY SW Dummy Read Data Register 1 (MMDC\_MPSWDRDR1)

Address: 21B\_0000h base + 89Ch offset = 21B\_089Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DUM_RD1																															
W																																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

#### MMDC\_MPSWDRDR1 field descriptions

Field	Description
31–0 DUM_RD1	Dummy read data1. This field holds the second data that is read from the DDR during SW dummy read access (i.e when SW_DUMMY_RD = 1). This field is valid only when SW_DUMMY_RD is de-asserted

### 32.12.71 MMDC PHY SW Dummy Read Data Register 2 (MMDC\_MPSWDRDR2)

Address: 21B\_0000h base + 8A0h offset = 21B\_08A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DUM_RD2																															
W																																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

#### MMDC\_MPSWDRDR2 field descriptions

Field	Description
31–0 DUM_RD2	Dummy read data2. This field holds the third data that is read from the DDR during SW dummy read access (i.e when SW_DUMMY_RD = 1). This field is valid only when SW_DUMMY_RD is de-asserted.

### 32.12.72 MMDC PHY SW Dummy Read Data Register 3 (MMDC\_MPSWDRDR3)

Address: 21B\_0000h base + 8A4h offset = 21B\_08A4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DUM_RD3																															
W																																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

### MMDC\_MPSWDRDR3 field descriptions

Field	Description
31–0 DUM_RD3	Dummy read data3. This field holds the forth data that is read from the DDR during SW dummy read access (i.e when SW_DUMMY_RD = 1). This field is valid only when SW_DUMMY_RD is de-asrtd.

## 32.12.73 MMDC PHY SW Dummy Read Data Register 4 (MMDC\_MPSWDRDR4)

Address: 21B\_0000h base + 8A8h offset = 21B\_08A8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DUM_RD4																															
W																																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

### MMDC\_MPSWDRDR4 field descriptions

Field	Description
31–0 DUM_RD4	Dummy read data4. This field holds the fifth data (only in case of burst length 8 (BL =1 )) that is read from the DDR during SW dummy read access (i.e when SW_DUMMY_RD = 1). This field is valid only when SW_DUMMY_RD is de-asrtd.

## 32.12.74 MMDC PHY SW Dummy Read Data Register 5 (MMDC\_MPSWDRDR5)

Address: 21B\_0000h base + 8ACh offset = 21B\_08ACh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DUM_RD5																															
W																																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

### MMDC\_MPSWDRDR5 field descriptions

Field	Description
31–0 DUM_RD5	Dummy read data5. This field holds the sixth data (only in case of burst length 8 (BL =1 )) that is read from the DDR during SW dummy read access (i.e when SW_DUMMY_RD = 1). This field is valid only when SW_DUMMY_RD is de-asrtd.

### 32.12.75 MMDC PHY SW Dummy Read Data Register 6 (MMDC\_MPSWDRDR6)

Address: 21B\_0000h base + 8B0h offset = 21B\_08B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DUM_RD6																															
W																																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

#### MMDC\_MPSWDRDR6 field descriptions

Field	Description
31–0 DUM_RD6	Dummy read data6. This field holds the seventh data (only in case of burst length 8 (BL = 1 )) that is read from the DDR during SW dummy read access (i.e when SW_DUMMY_RD = 1). This field is valid only when SW_DUMMY_RD is de-asserted.

### 32.12.76 MMDC PHY SW Dummy Read Data Register 7 (MMDC\_MPSWDRDR7)

Address: 21B\_0000h base + 8B4h offset = 21B\_08B4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DUM_RD7																															
W																																
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

#### MMDC\_MPSWDRDR7 field descriptions

Field	Description
31–0 DUM_RD7	Dummy read data7. This field holds the eighth data (only in case of burst length 8 (BL = 1 )) that is read from the DDR during SW dummy read access (i.e when SW_DUMMY_RD = 1). This field is valid only when SW_DUMMY_RD is de-asserted.

## 32.12.77 MMDC PHY Measure Unit Register (MMDC\_MPMUR0)

Address: 21B\_0000h base + 8B8h offset = 21B\_08B8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								MU_UNIT_DEL_NUM							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				FRC_MSR	MU_BYP_EN	MU_BYP_VAL									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### MMDC\_MPMUR0 field descriptions

Field	Description
31–26 Reserved	This read-only field is reserved and always has the value 0.
25–16 MU_UNIT_DEL_NUM	Number of delay units measured per cycle. This field is used in debug mode and holds the number of delay units that were measured by the measure unit per DDR clock cycle. The delay-lines that are used in every calibration process use that number for generating the desired delay.
15–12 Reserved	This read-only field is reserved and always has the value 0.
11 FRC_MSR	Force measurement on delay-lines. When this bit is asserted then a measurement process will be performed, where at the completion of the process the delay-lines will issue the desired delay. Upon completion of the measurement process the measure unit and the delay-lines will return to functional mode. This bit is self cleared.  <b>NOTE:</b> This bit should be used only during manual (SW) calibration and not while the DDR is functional (being accessed). After initial calibration is done the hardware performs periodic measurements to track any operating conditions changes. Hence, force measurements (FRC_MSR) should not be used. See <a href="#">Calibration Process</a> for more information.  <b>NOTE:</b> User should make sure that there is no active accesses to/from DDR before asserting this bit.  0 No measurement is performed 1 Perform measurement process
10 MU_BYP_EN	Measure unit bypass enable. This field is used in debug mode and when it is asserted then the delay-lines will use the number of delay units that are indicated at MU_BYP_VAL, otherwise the delay-lines will use the number of delay units that was measured by the measurement unit and are indicated at MU_UNIT_DEL_NUM  0 The delay-lines use delay units as indicated at MU_UNIT_DEL_NUM. 1 The delay-lines use delay units as indicated at MU_BYPASS_VAL.
9–0 MU_BYP_VAL	Number of delay units for measurement bypass. This field is used in debug mode and holds the number of delay units that will be used by the delay-lines when MU_BYP_EN is asserted.



### 32.12.78 MMDC Write CA delay-line controller (MMDC\_MPWRCADL)

This register is used to add fine-tuning adjustment to the CA (command/Address of LPDDR2 bus) relative to the DDR clock

Address: 21B\_0000h base + 8BCh offset = 21B\_08BCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0												WR_CA9_ DEL		WR_CA8_ DEL	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WR_CA7_ DEL		WR_CA6_ DEL		WR_CA5_ DEL		WR_CA4_ DEL		WR_CA3_ DEL		WR_CA2_ DEL		WR_CA1_ DEL		WR_CA0_ DEL	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### MMDC\_MPWRCADL field descriptions

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–18 WR_CA9_DEL	CA (Command/Address LPDDR2 bus) bit 9 delay fine tuning. This field holds the number of delay units that are added to CA (Command/Address bus) bit 9 relative to the clock.  00 No change in CA9 delay 01 Add CA9 delay of 1 delay unit 10 Add CA9 delay of 2 delay units. 11 Add CA9 delay of 3 delay units.
17–16 WR_CA8_DEL	CA (Command/Address LPDDR2 bus) bit 8 delay fine tuning. This field holds the number of delay units that are added to CA (Command/Address bus) bit 8 relative to the clock.  00 No change in CA8 delay 01 Add CA8 delay of 1 delay unit 10 Add CA8 delay of 2 delay units. 11 Add CA8 delay of 3 delay units.
15–14 WR_CA7_DEL	CA (Command/Address LPDDR2 bus) bit 7 delay fine tuning. This field holds the number of delay units that are added to CA (Command/Address bus) bit 7 relative to the clock.  00 No change in CA7 delay 01 Add CA7 delay of 1 delay unit 10 Add CA7 delay of 2 delay units. 11 Add CA7 delay of 3 delay units.
13–12 WR_CA6_DEL	CA (Command/Address LPDDR2 bus) bit 6 delay fine tuning. This field holds the number of delay units that are added to CA (Command/Address bus) bit 6 relative to the clock.  00 No change in CA6 delay

Table continues on the next page...

### MMDC\_MPWRCADL field descriptions (continued)

Field	Description
	01 Add CA6 delay of 1 delay unit 10 Add CA6 delay of 2 delay units. 11 Add CA6 delay of 3 delay units.
11–10 WR_CA5_DEL	CA (Command/Address LPDDR2 bus) bit 5 delay fine tuning. This field holds the number of delay units that are added to CA (Command/Address bus) bit 5 relative to the clock.  00 No change in CA5 delay 01 Add CA5 delay of 1 delay unit 10 Add CA5 delay of 2 delay units. 11 Add CA5 delay of 3 delay units.
9–8 WR_CA4_DEL	CA (Command/Address LPDDR2 bus) bit 4 delay fine tuning. This field holds the number of delay units that are added to CA (Command/Address bus) bit 4 relative to the clock.  00 No change in CA4 delay 01 Add CA4 delay of 1 delay unit 10 Add CA4 delay of 2 delay units. 11 Add CA4 delay of 3 delay units.
7–6 WR_CA3_DEL	CA (Command/Address LPDDR2 bus) bit 3 delay fine tuning. This field holds the number of delay units that are added to CA (Command/Address bus) bit 3 relative to the clock.  00 No change in CA3 delay 01 Add CA3 delay of 1 delay unit 10 Add CA3 delay of 2 delay units. 11 Add CA3 delay of 3 delay units.
5–4 WR_CA2_DEL	CA (Command/Address LPDDR2 bus) bit 2 delay fine tuning. This field holds the number of delay units that are added to CA (Command/Address bus) bit 2 relative to the clock.  00 No change in CA2 delay 01 Add CA2 delay of 1 delay unit 10 Add CA2 delay of 2 delay units. 11 Add CA2 delay of 3 delay units.
3–2 WR_CA1_DEL	CA (Command/Address LPDDR2 bus) bit 1 delay fine tuning. This field holds the number of delay units that are added to CA (Command/Address bus) bit 1 relative to the clock.  00 No change in CA1 delay 01 Add CA1 delay of 1 delay unit 10 Add CA1 delay of 2 delay units. 11 Add CA1 delay of 3 delay units.
1–0 WR_CA0_DEL	CA (Command/Address LPDDR2 bus) bit 0 delay fine tuning. This field holds the number of delay units that are added to CA (Command/Address bus) bit 0 relative to the clock.  00 No change in CA0 delay 01 Add CA0 delay of 1 delay unit 10 Add CA0 delay of 2 delay units. 11 Add CA0 delay of 3 delay units.

### 32.12.79 MMDC Duty Cycle Control Register (MMDC\_MPDCCR)

This register is used to control the duty cycle of the DQS and the primary clock (CK0) . Programming of that register is permitted by entering the DDR device into self-refresh mode through LPMD/DVFS mechanism

Address: 21B\_0000h base + 8C0h offset = 21B\_08C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	RD_DQS3_FT_DCC			RD_DQS2_FT_DCC			RD_DQS1_FT_DCC			RD_DQS0_FT_DCC			CK_FT1_DCC		
W																
Reset	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	CK_FT0_DCC			WR_DQS3_FT_DCC			WR_DQS2_FT_DCC			WR_DQS1_FT_DCC			WR_DQS0_FT_DCC		
W																
Reset	0	0	1	0	0	1	0	0	1	0	0	1	0	0	1	0

#### MMDC\_MPDCCR field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30–28 RD_DQS3_FT_DCC	Read DQS duty cycle fine tuning control of Byte3. This field controls the duty cycle of read DQS of Byte3 Note all the other options are not allowed 001 48.5% low 51.5% high 010 50% duty cycle (default) 100 51.5% low 48.5% high
27–25 RD_DQS2_FT_DCC	Read DQS duty cycle fine tuning control of Byte2. This field controls the duty cycle of read DQS of Byte2 Note all the other options are not allowed 001 48.5% low 51.5% high 010 50% duty cycle (default) 100 51.5% low 48.5% high
24–22 RD_DQS1_FT_DCC	Read DQS duty cycle fine tuning control of Byte1. This field controls the duty cycle of read DQS of Byte1 Note all the other options are not allowed 001 48.5% low 51.5% high 010 50% duty cycle (default) 100 51.5% low 48.5% high
21–19 RD_DQS0_FT_DCC	Read DQS duty cycle fine tuning control of Byte0. This field controls the duty cycle of read DQS of Byte0 Note all the other options are not allowed

Table continues on the next page...

### MMDC\_MPDCCR field descriptions (continued)

Field	Description
	001 48.5% low 51.5% high 010 50% duty cycle (default) 100 51.5% low 48.5% high
18–16 CK_FT1_DCC	Secondary duty cycle fine tuning control of DDR clock. This field controls the duty cycle of the DDR clock and is cascaded to CK_FT0_DCC Note all the other options are not allowed 001 48.5% low 51.5% high 010 50% duty cycle (default) 100 51.5% low 48.5% high
15 Reserved	This read-only field is reserved and always has the value 0.
14–12 CK_FT0_DCC	Primary duty cycle fine tuning control of DDR clock. This field controls the duty cycle of the DDR clock Note all the other options are not allowed 001 48.5% low 51.5% high 010 50% duty cycle (default) 100 51.5% low 48.5% high
11–9 WR_DQS3_FT_DCC	Write DQS duty cycle fine tuning control of Byte0. This field controls the duty cycle of write DQS of Byte0 Note all the other options are not allowed 001 48.5% low 51.5% high 010 50% duty cycle (default) 100 51.5% low 48.5% high
8–6 WR_DQS2_FT_DCC	Write DQS duty cycle fine tuning control of Byte1. This field controls the duty cycle of write DQS of Byte1 Note all the other options are not allowed 001 48.5% low 51.5% high 010 50% duty cycle (default) 100 51.5% low 48.5% high
5–3 WR_DQS1_FT_DCC	Write DQS duty cycle fine tuning control of Byte1. This field controls the duty cycle of write DQS of Byte1 Note all the other options are not allowed 001 48.5% low 51.5% high 010 50% duty cycle (default) 100 51.5% low 48.5% high
2–0 WR_DQS0_FT_DCC	Write DQS duty cycle fine tuning control of Byte0. This field controls the duty cycle of write DQS of Byte0 Note all the other options are not allowed 001 48.5% low 51.5% high 010 50% duty cycle (default) 100 51.5% low 48.5% high

# Chapter 33

## Network Interconnect Bus System (NIC-301)

### 33.1 Overview

This section provides an overview of the NIC-301 (Network Inter-Connect) AXI arbiter IP.

The NIC-301 (by ARM Ltd.) is a configurable AXI arbiter between several masters and slaves. The NIC-301 IP is designed so that many configuration options are selected at the hardware design stage, determined by SoC characteristics and needs, while several other configuration options are software-controlled.

This chapter covers in brief the NIC-301 functionality, while providing configuration details on the NIC-301 instances used in the chip. For complete details on the NIC-301 design, see the ARM specification, *AMBA® Network Interconnect (NIC-301) Technical Reference Manual*.

#### NOTE

The NIC-301 default settings are configured by Freescale's board support package (BSP), and in most cases should not be modified by the customer. The default settings have gone through exhaustive testing during the validation of the part, and have proven to work well for the part's intended target applications. Changes to the default settings may result in a degradation in system performance.

### 33.2 Block diagram

The NIC-301 AXI arbiter (or "CoreLink Network Interconnect") by ARM, provides configurable AXI-based interconnect logic, for connecting a number of masters (initiators) to several slaves (targets), via a configurable bus switches and bridging components.

The bus system is composed of three such instances: PER1, PER2, and FAST.

Each instance can include one or more bus switches, with additional logic.

This chapter provides details of the various instances and the selected configuration parameters.

The top level diagram of the bus system is shown in the following figure.

**Figure 33-1. NIC-301 Bus System**

**33.2.1 NIC-301 Main Features**

Key features of the NIC-301 module include the following:

- Address space memory mapping, including 'remap' functions.
- Programmer's view, for software-configured parameters, via "GPV" ports.
- Support for cross-clock domain synchronization.

**33.2.2 Modes and Operations**

The NIC-301 supports a normal functional mode only, as described in [Normal Mode](#).

**33.3 External Signals**

The NIC-301 has no external I/O interfaces.

**33.4 Memory Map and Register Definition**

This section includes the block memory map and detailed register descriptions.

Access to NIC-301 registers is provided through the global programmer's view (GPV) ports. Each GPV port provides access to the configuration registers of certain IP . The GPV base addresses are listed in the table below:

**Table 33-1. GPV ports memory allocations**

GPV	Associated NIC-301	System Bus	Chip Address	
			Start	End
GPV_0	FAST	Main DDR Arbitration	00B0_0000	00BF_FFFF

*Table continues on the next page...*

**Table 33-1. GPV ports memory allocations (continued)**

GPV	Associated NIC-301	System Bus	Chip Address	
			Start	End
GPV_1	PER1	Peripheral AHB/AXI Arbitration	00C0_0000	00CF_FFFF
GPV_2	PER2	Peripheral AHB/AXI Arbitration	00D0_0000	00DF_FFFF

### 33.4.1 Memory Map

The NIC-301 memory map, is dependent on the selected configuration option at time of creation.

A "template" map is provided in [Table 33-2](#) below. For specific features, refer to the configuration tables in [NIC Specific Parameters](#) to see which specific options are selected.

### 33.4.2 Configuration programmers model

The GPV's contain configuration registers, partitioned into a number of individual 4KB blocks.

The general structure of the registers is provided by the following tables:

- Address map of the programmers model, [Table 33-2](#)
- AMIB Registers, [Table 33-3](#)
- ASIB Registers, [Table 33-4](#)

**Table 33-2. Address map of the programmers model**

Address Offset from Base Address	Registers	Notes
0x000F_F000	Internal interface p registers	Maximum p = 61 Note <sup>1</sup>
	...	
0x000C_4000	Internal interface 2 registers	
0x000C_3000	Internal interface 1 registers	
0x000C_2000	Internal interface 0 registers	

*Table continues on the next page...*

**Table 33-2. Address map of the programmers model (continued)**

Address Offset from Base Address	Registers	Notes
0x000C_1000	Slave interface m registers	Maximum m = 127
	...	Note <sup>2</sup>
0x0004_4000	Slave interface 2 registers	
0x0004_3000	Slave interface 1 registers	
0x0004_2000	Slave interface 0 registers	
0x0004_1000	Master interface n registers	Maximum n = 63
	...	Note <sup>3</sup>
0x0000_4000	Master interface 2 registers	
0x0000_3000	Master interface 1 registers	
0x0000_2000	Master interface 0 registers	
0x0000_1000	ID registers	
0x0000_0000	Address control registers	Configurable base address <sup>4</sup>

1. Index refers to BI registers index
2. Index refer to ASIB registers index
3. Index refer to AMIB registers index
4. Reserved for internal use

### 33.4.2.1 Address control and ID registers

Registers at offsets 0x0–0xFFC are reserved for internal use.

### 33.4.2.2 AMBA master interface block (AMIB) configuration registers

The table below lists only the registers that affect the user. All other addresses are treated as "reserved".

**Table 33-3. AMIB Registers**

Offset	Register	Access	Width	Reset Value
0x024	fn_mod2 Bypass merge. This register is only present if upsizing or downsizing. See upsizing/downsizing data width functions in AMBA Network Interconnect TRM.	RW	1	0
0x040	wr_tidemark	RW	4	Note <sup>1</sup>

1. Reset value varies, default value chosen at RTL creation time is designed to suit normal operation.



### 33.4.2.3 ASIB (AMBA slave interface block) configuration registers

The table below lists only the registers that affect the user. All other addresses are treated as "reserved".

**Table 33-4. ASIB Registers**

Offset	Register	Access	Width	Reset Value
0x040	wr_tidemark Valid only for AXI slaves with WFIFO >=4.	RW	4	Note <sup>1</sup>
0x100	read_qos	RW	4	Note <sup>2</sup>
0x104	write_qos	RW	4	Note4

1. Reset value varies, default value chosen at RTL creation time is designed to suit typical operation cases.
2. QoS default is set at RTL creation time, and is listed in specific NIC-301 configuration tables in [NIC-specific parameters](#), as parameters "QoS qv\_value" in ASIB / AMIB parameter tables.

### 33.4.3 Register Descriptions

The NIC-301 registers are dependent upon the selected configuration, the type of ports, hardware-selected features and whether they have a GPV view. The addressing is associated to a specific port, by looking at the port's index number, under "apb\_slave" column.

The memory map template is provided in the [Configuration programmers model](#) above.

### 33.4.4 NIC Specific Parameters

This section details the configuration parameters of the NIC-301.

General notes:

1. The associated Master / Slave port interface ID (for each "GPV\_N") is specified by the "apb\_slave" / "apb\_master" <sup>1</sup> index. The Slave/Master interface, configuration registers start address is obtained by the following:

Start address (for "apb" index= "n") = GPV\_N base + (n \* 0x1000).

2. For the MX6FAST, QoS (Quality Of Service) is controlled by the QoS block and not by the GPV registers.
3. The security features of NIC-301 are not used, since master-slaves permissions are controlled by the CSU security policy/scheme.

---

1. "APB" stands for "ARM Peripheral Bus".

Table 33-5. NIC-301 parameters

Master	apb_slave	Tidemark	QOS qv_value	Comments
Configured via GPV_1 port				
DAP	72		2	
SDMA Burst	66		3	
SDMA Peripheral	67		3	
DCP	68		2	
CSI	73		2	
FEC	70		2	
USB02H	71		2	
uSDHC 1	74		2	
uSDHC 2	75		2	
uSDHC 3	76		2	
uSDHC 4	77		2	
Test Port	78	4	0	
Configured via GPV_0 port				
MPCore-0	66		QoS input control signal	
MPCore-1	67		QoS input control signal	
EPDC or SiPix (depends on fuse)	70	12	QoS input control signal	
LCDIF	72		QoS input control signal	
PXP	73	12	QoS input control signal	
GPU	74		QoS input control signal	

## Chapter 34

# On-Chip OTP Controller (OCOTP\_CTRL)

### 34.1 Overview

This section contains information describing the requirements for the on-chip eFuse OTP controller along with details about the block functionality and implementation.

In this document, the words "eFuse" and "OTP" are interchangeable. OCOTP refers to the hardware block itself.

#### 34.1.1 Features

The OCOTP provides the following features :

- 32-bit word restricted program and read to 2Kbits of eFuse OTP(128x8).
- Loading and housing of fuse content into shadow registers.
- memory-mapped (restricted) access to 2Kbits of shadow registers.
- Generation of hwv\_fuse (hardware visible fuse bus) and the hwv\_reg bus which is made of up of volatile PIO register based "fuses". The hwv\_reg bits come from the SCS (Software Controllable Signals) register.
- Generation of sticky\_reg which is consist of sticky register bits.
- Provide program-protect and read-protect efuse.
- Provide override and read protection of shadow register.
- CRC32 test for read-lock fuse content.

### 34.2 Clocks

The table found here describes the clock sources for OCOTP.

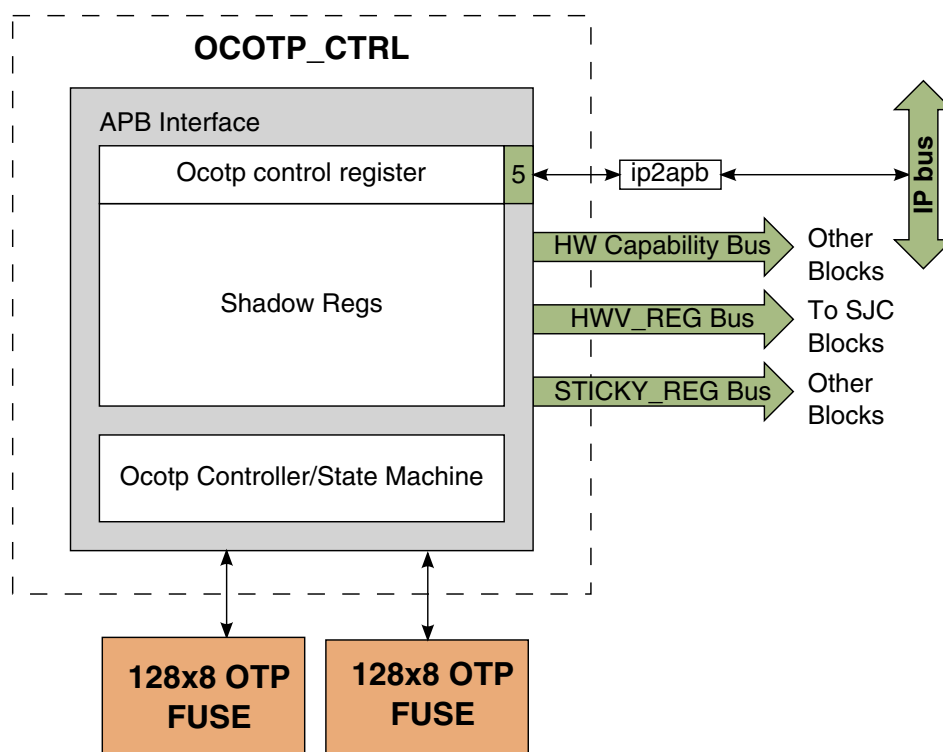
Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 34-1. OCOTP Clocks**

Clock name	Clock Root	Description
ipg_clk	ipg_clk_root	Peripheral clock
ipg_clk_s	ipg_clk_root	Peripheral access clock

## 34.3 Top-Level Symbol and Functional Overview

The figure found here shows the OCOTP system level diagram.



**Figure 34-1. OCOTP System Level Diagram**

### 34.3.1 Operation

The IP bus interface of the OCOTP provides two functions.

- Configure control registers for programming and reading fuse word.
- Override and read shadow registers.

For efuse, program can only be performed on bit and read is based on byte. OCOTP configuration for program and read are performed on 32-bit words for SW convenience. For writes, the 32-bit word reflects the "write-mask". Bit fields with 0 will not be programmed and bit fields with 1 will be programmed. OCOTP will program bit field with 1 in the fuse word one bit by one bit. For reads, OCOTP will read 4 times to get 4 bytes in the fuse word in order.

In this document, 2k bits fuse are divided into 8 banks by function. Each bank has 8 fuse words. In physical, 2k bits fuse are in two 128x8 efusebox.

### 34.3.1.1 Shadow Register Reload

All fuse words in efusebox are shadowed. Therefore, fuse information is available through memory mapped shadow registers. If fuses are subsequently programmed, the shadow registers should be reloaded to keep them coherent with the fuse bank arrays.

The "reload shadows" feature allows the user to force a reload of the shadow registers (including HW\_OCOTP\_LOCK) without having to reset the device. To force a reload, complete the following steps:

1. Set the HW\_OCOTP\_TIMING[STROBE\_READ] and HW\_OCOTP\_TIMING[RELAX] field value appropriately (as explained in a later section).
2. Check that HW\_OCOTP\_CTRL[BUSY] and HW\_OCOTP\_CTRL[ERROR] are clear. Overlapped accesses are not supported by the controller. Any pending write, read or reload must be completed before a new access can be requested.
3. Set the HW\_OCOTP\_CTRL[RELOAD\_SHADOWS] bit. OCOTP will read all the fuse one by one and put it into corresponding shadow register.
4. Wait for HW\_OCOTP\_CTRL[BUSY] and HW\_OCOTP\_CTRL[RELOAD\_SHADOWS] to be cleared by the controller.

The controller will automatically clear the HW\_OCOTP\_CTRL[RELOAD\_SHADOWS] bit after the successful completion of the operation.

### 34.3.1.2 Fuse and Shadow register read

All shadow registers are always readable through the APB bus except some secret keys regions. When their corresponding fuse lock bits are set, the shadow registers also become read locked. After read locking, reading from these registers will return 0xBADABADA.

In addition HW\_OCOTP\_CTRL[ERROR] will be set. It must be cleared by software before any new write, read or reload access can be issued. Subsequent reads to unlocked shadow locations will still work successfully however.

To read fuse word directly from fusebox correctly complete the following steps:

1. Program HW\_OCOTP\_TIMING[STROBE\_READ] and HW\_OCOTP\_TIMING[RELAX] fields with timing values to match the current frequency of the ipg\_clk. OTP read will work at maximum bus frequencies as long as the HW\_OCOTP\_TIMING parameters are set correctly.
2. Check that HW\_OCOTP\_CTRL[BUSY] and HW\_OCOTP\_CTRL[ERROR] are clear. Overlapped accesses are not supported by the controller. Any pending write, read or reload must be completed before a read access can be requested.
3. Write the requested address to HW\_OCOTP\_CTRL[ADDR].
4. Set HW\_OCOTP\_READ\_CTRL[READ\_FUSE] to 1. OCOTP will auto read 4 bytes in requested word address in fusebox one by one. Then put read value into HW\_OCOTP\_READ\_FUSE\_DATA register.
5. Once complete, the controller will clear BUSY. A read request to a protected or locked region will result in no OTP access and no setting of HW\_OCOTP\_CTRL[BUSY]. In addition HW\_OCOTP\_CTRL[ERROR] will be set. It must be cleared by software before any new access can be issued.
6. Read HW\_OCOTP\_READ\_FUSE\_DATA register to get fuse word value. HW\_OCOTP\_READ\_FUSE\_DATA will be 0xBADABADA when HW\_OCOTP\_CTRL[ERROR] is set.

### 34.3.1.3 Fuse and Shadow Register Writes

Shadow register bits can be overridden by software until the corresponding fuse lock bit for the region is set. When the lock shadow bit is set, the shadow registers for that lock region become write locked. The LOCK shadow register also has no shadow or fuse lock bits but it is always read only.

In order to avoid "rogue" code performing erroneous writes to OTP, a special unlocking sequence is required for writes to the fuse banks. To program fuse bank correctly complete the following steps:

1. Program HW\_OCOTP\_TIMING[STROBE\_PROG] and HW\_OCOTP\_TIMING[RELAX] fields with timing values to match the current frequency of the ipg\_clk. OTP writes will work at maximum bus frequencies as long as the HW\_OCOTP\_TIMING parameters are set correctly.

2. Check that HW\_OCOTP\_CTRL[BUSY] and HW\_OCOTP\_CTRL[ERROR] are clear. Overlapped accesses are not supported by the controller. Any pending write or reload must be completed before a write access can be requested.
3. Write the requested address to HW\_OCOTP\_CTRL[ADDR] and program the unlock code into HW\_OCOTP\_CTRL[WR\_UNLOCK]. This must be programmed for each write access. The lock code is documented in the register description. Both the unlock code and address can be written in the same operation.
4. Write the data to the HW\_OCOTP\_DATA register. This will automatically set HW\_OCOTP\_CTRL[BUSY] and clear HW\_OCOTP\_CTRL[WR\_UNLOCK]. To protect programming same OTP bit twice, before program OCOTP will automatically read fuse value in OTP and use read value to mask program data. The controller will use masked program data to program a 32-bit word in the OTP per the address in HW\_OCOTP\_CTRL[ADDR]. Bit fields with 1's will result in that OTP bit being programmed. Bit fields with 0's will be ignored. At the same time that the write is accepted, the controller makes an internal copy of HW\_OCOTP\_CTRL[ADDR] which cannot be updated until the next write sequence is initiated. This copy guarantees that erroneous writes to HW\_OCOTP\_CTRL[ADDR] will not affect an active write operation. It should also be noted that during the programming HW\_OCOTP\_DATA will shift right (with zero fill). This shifting is required to program the OTP serially. During the write operation, HW\_OCOTP\_DATA cannot be modified.
5. Once complete, the controller will clear BUSY. A write request to a protected or locked region will result in no OTP access and no setting of HW\_OCOTP\_CTRL[BUSY]. In addition HW\_OCOTP\_CTRL[ERROR] will be set. It must be cleared by software before any new write access can be issued.

It should be noted that write latencies to OTP are numbers of 10 micro-seconds per word. Write latencies will be based on amount of bit filed which is 1. For example : program half fuse bits in one word need 10us x 16.

For further details of OTP read/write operations see [eFUSE].

HW\_OCOTP\_CTRL[ERROR] will be set under the following conditions:

- A write is performed to a shadow register during a shadow reload (essentially, while HW\_OCOTP\_CTRL[RELOAD\_SHADOWS] is set. In addition, the contents of the shadow register shall not be updated.
- A write is performed to a shadow register which has been locked.
- A read is performed to from a shadow register which has been read locked.
- A program is performed to a fuse word which has been locked.
- A read is performed to from a fuse word which has been read locked.

### 34.3.1.4 Write Postamble

Due to internal electrical characteristics of the OTP during writes, all OTP operations following a write must be separated by 2 us after the clearing of HW\_OCOTP\_CTRL\_BUSY following the write. This guarantees programming voltages on-chip to reach a steady state when exiting a write sequence. This includes reads, shadow reloads, or other writes.

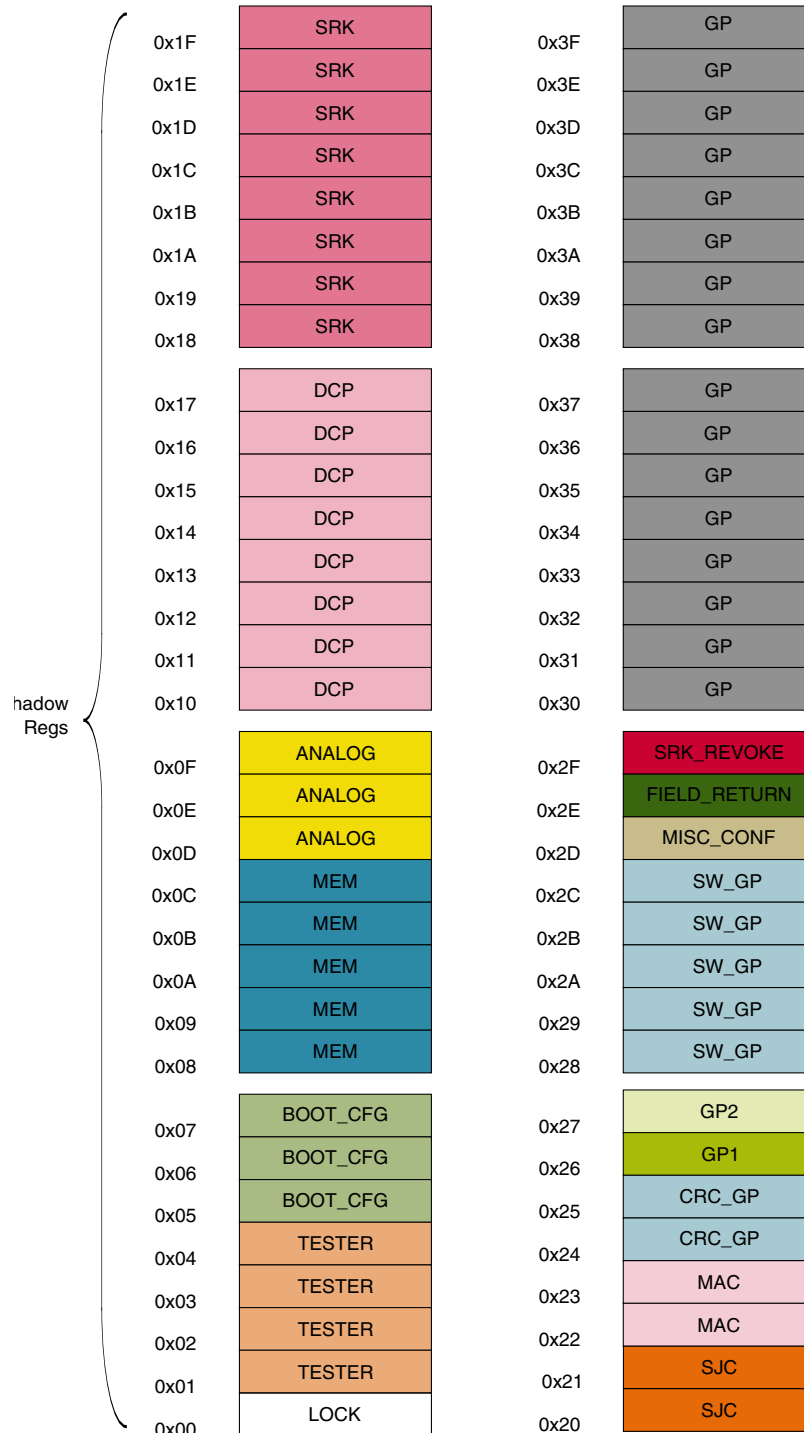
A recommended software sequence to meet the postamble requirements is as follows:

- Issue the write and poll for BUSY (as per [Fuse Shadow Memory Footprint](#)).
- Once BUSY is clear, use HW\_DIGCTL\_MICROSECONDS to wait 2 us.
- Perform the next OTP operation.

### 34.3.2 Fuse Shadow Memory Footprint

The OTP memory footprint is shown in the figure below. The registers are grouped by lock region. Their names correspond to the PIO register and fusemap names.





### 34.3.3 OTP Read/Write Timing Parameters

There are 3 timing fields contained in the HW\_OCOTP\_TIMING register that specify counter limit values, which are used to time how long the state machine remains in the various states, as well as specify the STROBE signal timing.

They are all specified in ipg\_clk cycles. Since the ipg\_clk frequency can be set to a range of values, these parameters must be adjusted with the clock to yield the appropriate delay.

The HW\_OCOTP\_TIMING[RELAX] field specifies how long to remain in states to meet setup and hold timing requirement in fuse spec. This parameter should be set by the following equation:

$$t_{RELAX} = t_{HP\_PG} = (HW\_OCOTP\_TIMING[RELAX]+1)/ipg\_frequency > 16.2ns$$

HW\_OCOTP\_TIMING[RELAX] field is used to create other setup and hold timing delays in addition to tHP\_PG. For all timing to be met, this is the max delay that must be programmed.

Except for setup and hold timing delay, there are 2 timing parameters for STROBE signal pulse width in program and read.

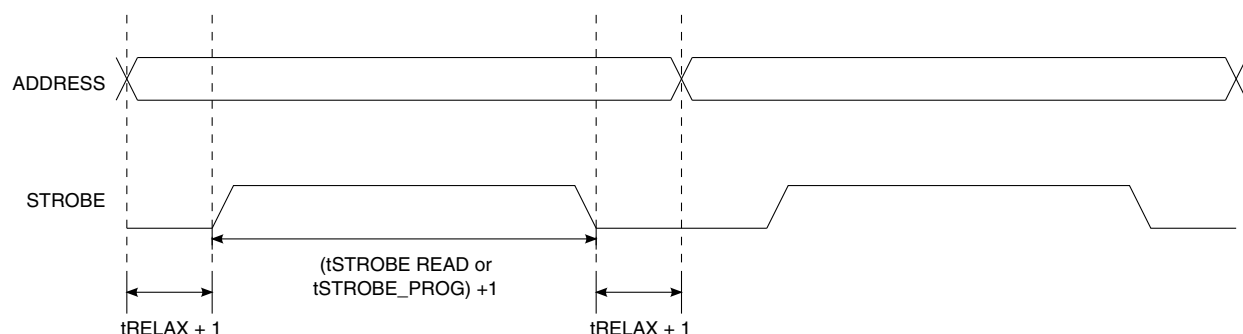
The HW\_OCOTP\_TIMING[STROBE\_PROG] field specifies the period of the STROBE signal for fuse writes and is given in units of ipg\_clk cycles. This value should be specified so that the requirement for the time when the STROBE signal is asserted high is met:  $9000ns < t_{PGM} < 11000ns$  is met. Even though a range is given for tPGM, it is advised in [eFUSE] to program for a value of 10000ns. Therefore, this field should be set according to the following equation:

$$t_{PGM} = ((HW\_OCOTP\_TIMING[STROBE\_PROG]+1) - 2*(HW\_OCOTP\_TIMING[RELAX]+1))/ipg\_frequency = 10000ns.$$

The HW\_OCOTP\_TIMING[STROBE\_READ] field specifies the period of the STROBE signal for fuse reads and is given in units of ipg\_clk cycles. This field should be set according to the following equation:

$$t_{RD} = ((HW\_OCOTP\_TIMING[STROBE\_READ]+1) - 2*(HW\_OCOTP\_TIMING[RELAX]+1))/ipg\_frequency > 36ns.$$

The figure below illustrates the relationship between the STROBE signal in programming and reading mode, as well as the timing PIO register fields that affect it. The implementation uses one counter to generate the STROBE waveform within one period and a second counter counts the number of cycles to create for programming the designated word.



**Figure 34-3. STROBE Signal Creation and Timing**

### 34.3.4 Hardware Visible Fuses

The `hwv_fuse` bus emanates from the OCOTP block and goes to various other blocks inside the chip. This bus is made up of the shadow register bits for banks 0, 1, 2 and 4.

Only a subset of these fuse bits are currently used by the hardware. The fuse bits are initially copied from the eFuse banks after reset is deasserted. When all fuse bits are loaded into their shadow registers, the OCOTP asserts the `fuse_latched` output signal.

The `hwv_reg` bus also comes from the OCOTP. Its source is the `HW_OCOTP_SCS` register. This register has 1 defined bit, the `HAB_JDE` bit, that is connected to the SJC block. The SCS bits are intended to be used as volatile fuse bits under software control. Additional bits will be defined as needed in future implementations.

The system-wide reset sequence must be coordinated by the system reset controller so that the `hwv_fuse` and `hwv_reg` busses are stable and reflect the values of the fuses before they are used by the rest of the system.

### 34.3.5 Behavior During Reset

The OCOTP is always active. The shadow registers automatically load the appropriate OTP contents after reset is deasserted. During this load-time `HW_OCOTP_CTRL_BUSY` is set. The load time is similar to that of a "reload shadow" operation.

### 34.3.6 Secure JTAG control

The JTAG control fuses are used to allow or disallow JTAG access to secured resources.

Three JTAG security levels are envisioned, as shown in the table below.

**Table 34-2. JTAG Security Level Control Bits**

Security Mode	JTAG_SMODE	Description
No Debug	2'b11	The highest security level.
Secure JTAG	2'b01	Limit the JTAG access by using key based authentication mechanism.
JTAG Enable	2'b00	Low Security, all JTAG features are enabled.

## 34.4 Fuse Map

See your Freescale representative for the fuse map (fuse bit definition).

## 34.5 OCOTP Memory Map/Register Definition

OCOTP Hardware Register Format Summary

**OCOTP memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
21B_C000	OTP Controller Control Register (OCOTP_CTRL)	32	R/W	0000_0000h	<a href="#">34.5.1/2188</a>
21B_C004	OTP Controller Control Register (OCOTP_CTRL_SET)	32	R/W	0000_0000h	<a href="#">34.5.1/2188</a>
21B_C008	OTP Controller Control Register (OCOTP_CTRL_CLR)	32	R/W	0000_0000h	<a href="#">34.5.1/2188</a>
21B_C00C	OTP Controller Control Register (OCOTP_CTRL_TOG)	32	R/W	0000_0000h	<a href="#">34.5.1/2188</a>
21B_C010	OTP Controller Timing Register (OCOTP_TIMING)	32	R/W	0146_1299h	<a href="#">34.5.2/2190</a>
21B_C020	OTP Controller Write Data Register (OCOTP_DATA)	32	R/W	0000_0000h	<a href="#">34.5.3/2190</a>
21B_C030	OTP Controller Write Data Register (OCOTP_READ_CTRL)	32	R/W	0000_0000h	<a href="#">34.5.4/2191</a>
21B_C040	OTP Controller Read Data Register (OCOTP_READ_FUSE_DATA)	32	R/W	0000_0000h	<a href="#">34.5.5/2192</a>
21B_C050	Sticky bit Register (OCOTP_SW_STICKY)	32	R/W	0000_0000h	<a href="#">34.5.6/2192</a>
21B_C060	Software Controllable Signals Register (OCOTP_SCS)	32	R/W	0000_0000h	<a href="#">34.5.7/2194</a>
21B_C064	Software Controllable Signals Register (OCOTP_SCS_SET)	32	R/W	0000_0000h	<a href="#">34.5.7/2194</a>
21B_C068	Software Controllable Signals Register (OCOTP_SCS_CLR)	32	R/W	0000_0000h	<a href="#">34.5.7/2194</a>
21B_C06C	Software Controllable Signals Register (OCOTP_SCS_TOG)	32	R/W	0000_0000h	<a href="#">34.5.7/2194</a>
21B_C070	OTP Controller CRC test address (OCOTP_CRC_ADDR)	32	R/W	0000_0000h	<a href="#">34.5.8/2195</a>

*Table continues on the next page...*

## OCOTP memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
21B_C080	OTP Controller CRC Value Register (OCOTP_CRC_VALUE)	32	R/W	0000_0000h	<a href="#">34.5.9/2195</a>
21B_C090	OTP Controller Timing Register (OCOTP_UMC_TIMING)	32	R/W	0000_0042h	<a href="#">34.5.10/2196</a>
21B_C0A0	OTP Controller Version Register (OCOTP_VERSION)	32	R/W	0300_0000h	<a href="#">34.5.11/2196</a>
21B_C400	Value of OTP Bank0 Word0 (Lock controls) (OCOTP_LOCK)	32	R/W	0000_0000h	<a href="#">34.5.12/2197</a>
21B_C410	Value of OTP Bank0 Word1 (Configuration and Manufacturing Info.) (OCOTP_CFG0)	32	R/W	0000_0000h	<a href="#">34.5.13/2200</a>
21B_C420	Value of OTP Bank0 Word2 (Configuration and Manufacturing Info.) (OCOTP_CFG1)	32	R/W	0000_0000h	<a href="#">34.5.14/2200</a>
21B_C430	Value of OTP Bank0 Word3 (Configuration and Manufacturing Info.) (OCOTP_CFG2)	32	R/W	0000_0000h	<a href="#">34.5.15/2201</a>
21B_C440	Value of OTP Bank0 Word4 (Configuration and Manufacturing Info.) (OCOTP_CFG3)	32	R/W	0000_0000h	<a href="#">34.5.16/2201</a>
21B_C450	Value of OTP Bank0 Word5 (Configuration and Manufacturing Info.) (OCOTP_CFG4)	32	R/W	0000_0000h	<a href="#">34.5.17/2202</a>
21B_C460	Value of OTP Bank0 Word6 (Configuration and Manufacturing Info.) (OCOTP_CFG5)	32	R/W	0000_0000h	<a href="#">34.5.18/2202</a>
21B_C470	Value of OTP Bank0 Word7 (Configuration and Manufacturing Info.) (OCOTP_CFG6)	32	R/W	0000_0000h	<a href="#">34.5.19/2203</a>
21B_C480	Value of OTP Bank1 Word0 (Memory Related Info.) (OCOTP_MEM0)	32	R/W	0000_0000h	<a href="#">34.5.20/2203</a>
21B_C490	Value of OTP Bank1 Word1 (Memory Related Info.) (OCOTP_MEM1)	32	R/W	0000_0000h	<a href="#">34.5.21/2204</a>
21B_C4A0	Value of OTP Bank1 Word2 (Memory Related Info.) (OCOTP_MEM2)	32	R/W	0000_0000h	<a href="#">34.5.22/2204</a>
21B_C4B0	Value of OTP Bank1 Word3 (Memory Related Info.) (OCOTP_MEM3)	32	R/W	0000_0000h	<a href="#">34.5.23/2205</a>
21B_C4C0	Value of OTP Bank1 Word4 (Memory Related Info.) (OCOTP_MEM4)	32	R/W	0000_0000h	<a href="#">34.5.24/2205</a>
21B_C4D0	Value of OTP Bank1 Word5 (Memory Related Info.) (OCOTP_ANA0)	32	R/W	0000_0000h	<a href="#">34.5.25/2206</a>
21B_C4E0	Value of OTP Bank1 Word6 (General Purpose Customer Defined Info.) (OCOTP_ANA1)	32	R/W	0000_0000h	<a href="#">34.5.26/2206</a>
21B_C4F0	Value of OTP Bank1 Word7 (General Purpose Customer Defined Info.) (OCOTP_ANA2)	32	R/W	0000_0000h	<a href="#">34.5.27/2207</a>
21B_C500	Shadow Register for OTP Bank2 Word0 (DCP and CRYPTO Key) (OCOTP_DCP0)	32	R/W	0000_0000h	<a href="#">34.5.28/2207</a>
21B_C510	Shadow Register for OTP Bank2 Word1 (DCP and CRYPTO Key) (OCOTP_DCP1)	32	R/W	0000_0000h	<a href="#">34.5.29/2208</a>
21B_C520	Shadow Register for OTP Bank2 Word2 (DCP and CRYPTO Key) (OCOTP_DCP2)	32	R/W	0000_0000h	<a href="#">34.5.30/2208</a>

Table continues on the next page...

## OCOTP memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
21B_C530	Shadow Register for OTP Bank2 Word3 (DCP and CRYPTO Key) (OCOTP_DCP3)	32	R/W	0000_0000h	<a href="#">34.5.31/2209</a>
21B_C540	Shadow Register for OTP Bank2 Word4 (DCP Key) (OCOTP_DCP4)	32	R/W	0000_0000h	<a href="#">34.5.32/2209</a>
21B_C550	Shadow Register for OTP Bank2 Word5 (DCP Key) (OCOTP_DCP5)	32	R/W	0000_0000h	<a href="#">34.5.33/2210</a>
21B_C560	Shadow Register for OTP Bank2 Word6 (DCP Key) (OCOTP_DCP6)	32	R/W	0000_0000h	<a href="#">34.5.34/2210</a>
21B_C570	Shadow Register for OTP Bank2 Word7 (DCP Key) (OCOTP_DCP7)	32	R/W	0000_0000h	<a href="#">34.5.35/2211</a>
21B_C580	Shadow Register for OTP Bank3 Word0 (SRK Hash) (OCOTP_SRK0)	32	R/W	0000_0000h	<a href="#">34.5.36/2211</a>
21B_C590	Shadow Register for OTP Bank3 Word1 (SRK Hash) (OCOTP_SRK1)	32	R/W	0000_0000h	<a href="#">34.5.37/2212</a>
21B_C5A0	Shadow Register for OTP Bank3 Word2 (SRK Hash) (OCOTP_SRK2)	32	R/W	0000_0000h	<a href="#">34.5.38/2212</a>
21B_C5B0	Shadow Register for OTP Bank3 Word3 (SRK Hash) (OCOTP_SRK3)	32	R/W	0000_0000h	<a href="#">34.5.39/2213</a>
21B_C5C0	Shadow Register for OTP Bank3 Word4 (SRK Hash) (OCOTP_SRK4)	32	R/W	0000_0000h	<a href="#">34.5.40/2213</a>
21B_C5D0	Shadow Register for OTP Bank3 Word5 (SRK Hash) (OCOTP_SRK5)	32	R/W	0000_0000h	<a href="#">34.5.41/2214</a>
21B_C5E0	Shadow Register for OTP Bank3 Word6 (SRK Hash) (OCOTP_SRK6)	32	R/W	0000_0000h	<a href="#">34.5.42/2214</a>
21B_C5F0	Shadow Register for OTP Bank3 Word7 (SRK Hash) (OCOTP_SRK7)	32	R/W	0000_0000h	<a href="#">34.5.43/2215</a>
21B_C600	Value of OTP Bank4 Word0 (Secure JTAG Response Field) (OCOTP_SJC_RESP0)	32	R/W	0000_0000h	<a href="#">34.5.44/2215</a>
21B_C610	Value of OTP Bank4 Word1 (Secure JTAG Response Field) (OCOTP_SJC_RESP1)	32	R/W	0000_0000h	<a href="#">34.5.45/2216</a>
21B_C620	Value of OTP Bank4 Word2 (MAC Address) (OCOTP_MAC0)	32	R/W	0000_0000h	<a href="#">34.5.46/2217</a>
21B_C630	Value of OTP Bank4 Word3 (MAC Address) (OCOTP_MAC1)	32	R/W	0000_0000h	<a href="#">34.5.47/2217</a>
21B_C640	Value of OTP Bank4 Word4 (HW Capabilities) (OCOTP_CRC0)	32	R/W	0000_0000h	<a href="#">34.5.48/2218</a>
21B_C650	Value of OTP Bank4 Word5 (HW Capabilities) (OCOTP_CRC1)	32	R/W	0000_0000h	<a href="#">34.5.49/2218</a>
21B_C660	Value of OTP Bank4 Word6 (HW Capabilities) (OCOTP_GP1)	32	R/W	0000_0000h	<a href="#">34.5.50/2219</a>
21B_C670	Value of OTP Bank4 Word7 (HW Capabilities) (OCOTP_GP2)	32	R/W	0000_0000h	<a href="#">34.5.51/2219</a>
21B_C680	Value of OTP Bank5 Word0 (HW Capabilities) (OCOTP_SW_GP0)	32	R/W	0000_0000h	<a href="#">34.5.52/2220</a>

Table continues on the next page...

## OCOTP memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
21B_C690	Value of OTP Bank5 Word1 (HW Capabilities) (OCOTP_SW_GP1)	32	R/W	0000_0000h	<a href="#">34.5.53/2220</a>
21B_C6A0	Value of OTP Bank5 Word2 (HW Capabilities) (OCOTP_SW_GP2)	32	R/W	0000_0000h	<a href="#">34.5.54/2221</a>
21B_C6B0	Value of OTP Bank5 Word3 (HW Capabilities) (OCOTP_SW_GP3)	32	R/W	0000_0000h	<a href="#">34.5.55/2221</a>
21B_C6C0	Value of OTP Bank5 Word4 (HW Capabilities) (OCOTP_SW_GP4)	32	R/W	0000_0000h	<a href="#">34.5.56/2222</a>
21B_C6D0	Value of OTP Bank5 Word5 (HW Capabilities) (OCOTP_MISC_CONF)	32	R/W	0000_0000h	<a href="#">34.5.57/2222</a>
21B_C6E0	Value of OTP Bank5 Word6 (HW Capabilities) (OCOTP_FIELD_RETURN)	32	R/W	0000_0000h	<a href="#">34.5.58/2223</a>
21B_C6F0	Value of OTP Bank5 Word7 (HW Capabilities) (OCOTP_SRK_REVOKE)	32	R/W	0000_0000h	<a href="#">34.5.59/2223</a>
21B_C700	Value of OTP Bank6 Word0 (HW Capabilities) (OCOTP_GP_LO0)	32	R/W	0000_0000h	<a href="#">34.5.60/2224</a>
21B_C710	Value of OTP Bank6 Word1 (HW Capabilities) (OCOTP_GP_LO1)	32	R/W	0000_0000h	<a href="#">34.5.61/2224</a>
21B_C720	Value of OTP Bank6 Word2 (HW Capabilities) (OCOTP_GP_LO2)	32	R/W	0000_0000h	<a href="#">34.5.62/2225</a>
21B_C730	Value of OTP Bank6 Word3 (HW Capabilities) (OCOTP_GP_LO3)	32	R/W	0000_0000h	<a href="#">34.5.63/2225</a>
21B_C740	Value of OTP Bank6 Word4 (HW Capabilities) (OCOTP_GP_LO4)	32	R/W	0000_0000h	<a href="#">34.5.64/2226</a>
21B_C750	Value of OTP Bank6 Word5 (HW Capabilities) (OCOTP_GP_LO5)	32	R/W	0000_0000h	<a href="#">34.5.65/2226</a>
21B_C760	Value of OTP Bank6 Word6 (HW Capabilities) (OCOTP_GP_LO6)	32	R/W	0000_0000h	<a href="#">34.5.66/2227</a>
21B_C770	Value of OTP Bank6 Word7 (HW Capabilities) (OCOTP_GP_LO7)	32	R/W	0000_0000h	<a href="#">34.5.67/2227</a>
21B_C780	Value of OTP Bank7 Word0 (HW Capabilities) (OCOTP_GP_HI0)	32	R/W	0000_0000h	<a href="#">34.5.68/2228</a>
21B_C790	Value of OTP Bank7 Word1 (HW Capabilities) (OCOTP_GP_HI1)	32	R/W	0000_0000h	<a href="#">34.5.69/2228</a>
21B_C7A0	Value of OTP Bank7 Word2 (HW Capabilities) (OCOTP_GP_HI2)	32	R/W	0000_0000h	<a href="#">34.5.70/2229</a>
21B_C7B0	Value of OTP Bank7 Word3 (HW Capabilities) (OCOTP_GP_HI3)	32	R/W	0000_0000h	<a href="#">34.5.71/2229</a>
21B_C7C0	Value of OTP Bank7 Word4 (HW Capabilities) (OCOTP_GP_HI4)	32	R/W	0000_0000h	<a href="#">34.5.72/2230</a>
21B_C7D0	Value of OTP Bank7 Word5 (HW Capabilities) (OCOTP_GP_HI5)	32	R/W	0000_0000h	<a href="#">34.5.73/2230</a>
21B_C7E0	Value of OTP Bank7 Word6 (HW Capabilities) (OCOTP_GP_HI6)	32	R/W	0000_0000h	<a href="#">34.5.74/2231</a>

Table continues on the next page...

## OCOTP memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
21B_C7F0	Value of OTP Bank7 Word7 (HW Capabilities) (OCOTP_GP_HI7)	32	R/W	0000_0000h	<a href="#">34.5.75/2231</a>

### 34.5.1 OTP Controller Control Register (OCOTP\_CTRLn)

The OCOTP Control and Status Register specifies the copy state, as well as the control required for random access of the OTP memory

OCOTP\_CTRL: 0x000

OCOTP\_CTRL\_SET: 0x004

OCOTP\_CTRL\_CLR: 0x008

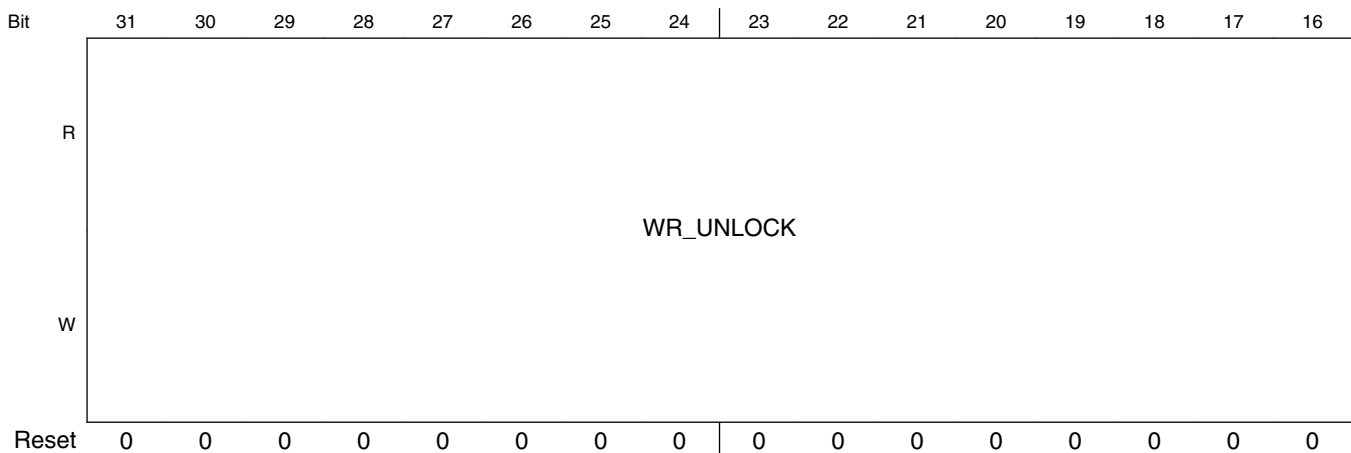
OCOTP\_CTRL\_TOG: 0x00C

The OCOTP Control and Status Register provides the necessary software interface for performing read and write operations to the On-Chip OTP (One-Time Programmable ROM). The control fields such as WR\_UNLOCK, ADDR and BUSY/ERROR may be used in conjunction with the OCOTP\_DATA register to perform write operations. Read operations to the On-Chip OTP are involving ADDR, BUSY/ERROR bit field and OCOTP\_READ\_CTRL register. Read value is saved in OCOTP\_READ\_FUSE\_DATA register.

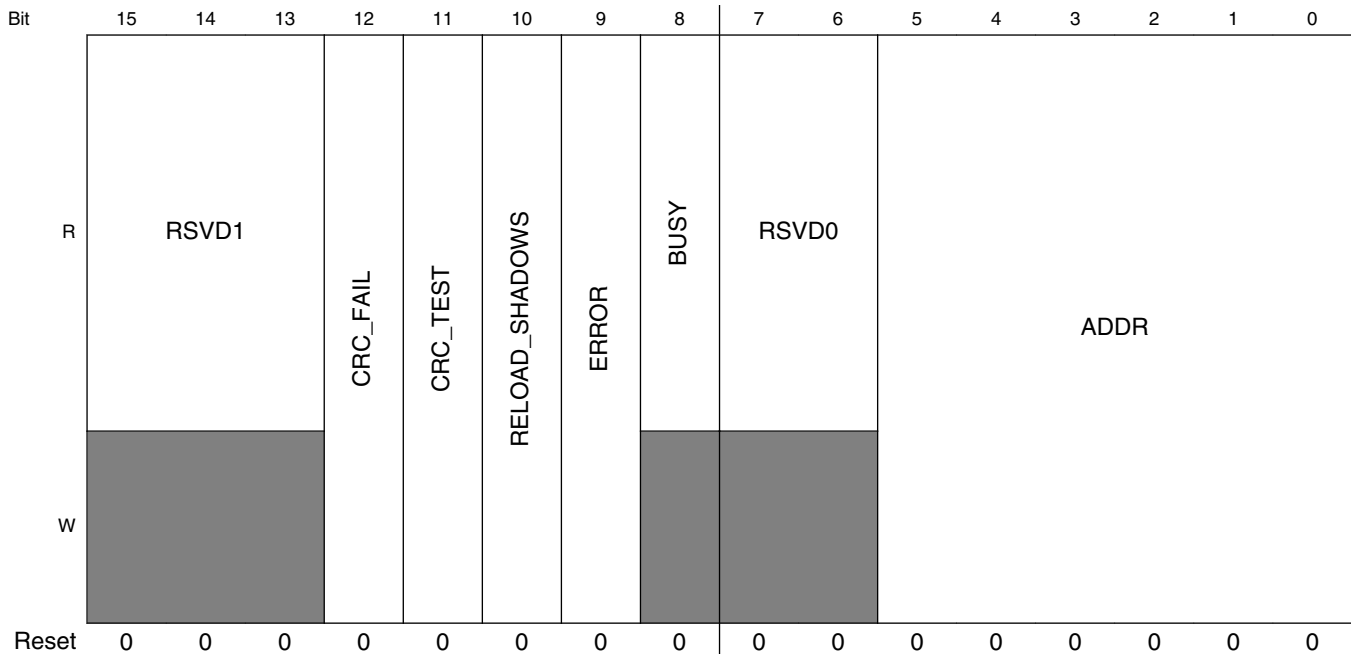
#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 0h offset + (4d × i), where i=0d to 3d







OCOTP\_CTRLn field descriptions

Field	Description
31–16 WR_UNLOCK	Write 0x3E77 to enable OTP write accesses. NOTE: This register must be unlocked on a write-by-write basis (a write is initiated when HW_OCOTP_DATA is written), so the UNLOCK bitfield must contain the correct key value during all writes to HW_OCOTP_DATA, otherwise a write shall not be initiated. This field is automatically cleared after a successful write completion (clearing of BUSY).  0x3E77 <b>KEY</b> — Key needed to unlock HW_OCOTP_DATA register.
15–13 RSVD1	Reserved
12 CRC_FAIL	Set by controller when calculated CRC value is not equal to appointed CRC fuse word
11 CRC_TEST	Set to calculate CRC according to start address and end address in CRC_ADDR register. And compare with CRC fuse word according CRC address in CRC_ADDR register to generate CRC_FAIL flag
10 RELOAD_SHADOWS	Set to force re-loading the shadow registers (HW/SW capability and LOCK). This operation will automatically set BUSY. Once the shadow registers have been re-loaded, BUSY and RELOAD_SHADOWS are automatically cleared by the controller.
9 ERROR	Set by the controller when an access to a locked region(OTP or shadow register) is requested. Must be cleared before any further access can be performed. This bit can only be set by the controller. This bit is also set if the Pin interface is active and software requests an access to the OTP. In this instance, the ERROR bit cannot be cleared until the Pin interface access has completed. Reset this bit by writing a one to the SCT clear address space and not by a general write.
8 BUSY	OTP controller status bit. When active, no new write access or read access to OTP(including RELOAD_SHADOWS) can be performed. Cleared by controller when access complete. After reset (or after setting RELOAD_SHADOWS), this bit is set by the controller until the HW/SW and LOCK registers are successfully copied, after which time it is automatically cleared by the controller.
7–6 RSVD0	Reserved
5–0 ADDR	OTP write and read access address register. Specifies one of 64 word address locations (0x00 - 0x3f). If a valid access is accepted by the controller, the controller makes an internal copy of this value. This internal copy will not update until the access is complete.

### 34.5.2 OTP Controller Timing Register (OCOTP\_TIMING)

The OCOTP Data Register is used for OTP Programming

This register specifies timing parameters for programming and reading the OCOTP fuse array.

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 10h offset = 21B\_C010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSRVD0				WAIT				STROBE_READ				RELAX				STROBE_PROG															
W																																
Reset	0	0	0	0	0	0	0	1	0	1	0	0	0	1	1	0	0	0	0	1	0	0	1	0	1	0	0	1	1	0	0	1

#### OCOTP\_TIMING field descriptions

Field	Description
31–28 RSRVD0	These bits always read back zero.
27–22 WAIT	This count value specifies time interval between auto read and write access in one time program. It is given in number of ipg_clk periods.
21–16 STROBE_READ	This count value specifies the strobe period in one time read OTP. $Trd = ((STROBE\_READ+1) - 2*(RELAX+1)) / ipg\_clk\_freq$ . It is given in number of ipg_clk periods.
15–12 RELAX	This count value specifies the time to add to all default timing parameters other than the Tpgm and Trd. It is given in number of ipg_clk periods.
11–0 STROBE_PROG	This count value specifies the strobe period in one time write OTP. $Tpgm = ((STROBE\_PROG+1) - 2*(RELAX+1)) / ipg\_clk\_freq$ . It is given in number of ipg_clk periods.

### 34.5.3 OTP Controller Write Data Register (OCOTP\_DATA)

The OCOTP Data Register is used for OTP Programming

This register is used in conjunction with OCOTP\_CTRL to perform one-time writes to the OTP. Please see the "Software Write Sequence" section for operating details.

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 20h offset = 21B\_C020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### OCOTP\_DATA field descriptions

Field	Description
31–0 DATA	Used to initiate a write to OTP. Please see the "Software Write Sequence" section for operating details.

## 34.5.4 OTP Controller Write Data Register (OCOTP\_READ\_CTRL)

The OCOTP Register is used for OTP Read

This register is used in conjunction with OCOTP\_CTRL to perform one time read to the OTP. Please see the "Software read Sequence" section for operating details.

### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 30h offset = 21B\_C030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RSVD0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD0															READ_FUSE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**OCOTP\_READ\_CTRL field descriptions**

Field	Description
31–1 RSVD0	Reserved
0 READ_FUSE	Used to initiate a read to OTP. Please see the "Software read Sequence" section for operating details.

### 34.5.5 OTP Controller Read Data Register (OCOTP\_READ\_FUSE\_DATA)

The OCOTP Data Register is used for OTP Read

The data read from OTP

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 40h offset = 21B\_C040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**OCOTP\_READ\_FUSE\_DATA field descriptions**

Field	Description
31–0 DATA	The data read from OTP

### 34.5.6 Sticky bit Register (OCOTP\_SW\_STICKY)

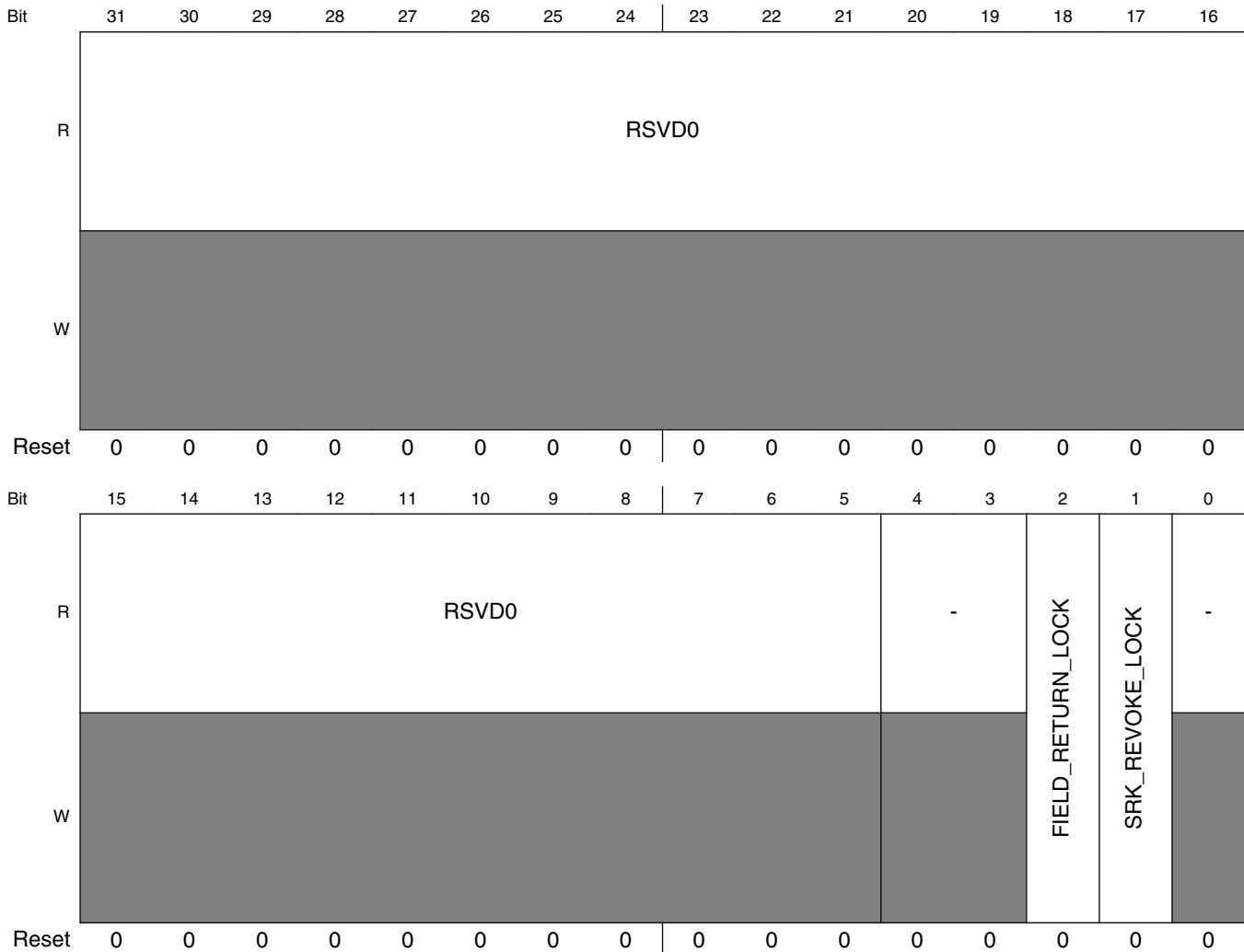
Some SW sticky bits .

Some sticky bits are used by SW to lock some fuse area, shadow registers and other features.

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 50h offset = 21B\_C050h

**OCOTP\_SW\_STICKY field descriptions**

Field	Description
31–5 RSVD0	Reserved
4–3 -	Reserved
2 FIELD_RETURN_LOCK	Shadow register write and OTP write lock for FIELD_RETURN region. When set, the writing of this region's shadow register and OTP fuse word are blocked. Once this bit is set, it is always high unless a POR is issued.
1 SRK_REVOKE_LOCK	Shadow register write and OTP write lock for SRK_REVOKE region. When set, the writing of this region's shadow register and OTP fuse word are blocked. Once this bit is set, it is always high unless a POR is issued.
0 -	Reserved

### 34.5.7 Software Controllable Signals Register (OCOTP\_SCSn)

OCOTP\_SCS: 0x060

OCOTP\_SCS\_SET: 0x064

OCOTP\_SCS\_CLR: 0x068

OCOTP\_SCS\_TOG: 0x06C

This register holds volatile configuration values that can be set and locked by trusted software. All values are returned to their default values after POR.

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 60h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	LOCK	SPARE														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SPARE															HAB_JDE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### OCOTP\_SCSn field descriptions

Field	Description
31 LOCK	When set, all of the bits in this register are locked and can not be changed through SW programming. This bit is only reset after a POR is issued.
30–1 SPARE	Unallocated read/write bits for implementation specific software use.
0 HAB_JDE	<p>HAB JTAG Debug Enable. This bit is used by the HAB to enable JTAG debugging, assuming that a properlay signed command to do so is found and validated by the HAB.</p> <p>The HAB must lock the register before passing control to the OS whether or not JTAG debugging has been enabled.</p> <p>Once JTAG is enabled by this bit, it can not be disabled unless the system is reset by POR. 0: JTAG debugging is not enabled by the HAB (it may still be enabled by other mechanisms).</p> <p>1: JTAG debugging is enabled by the HAB (though this signal may be gated off).</p>

### 34.5.8 OTP Controller CRC test address (OCOTP\_CRC\_ADDR)

The OCOTP Data Register is used for OTP Read

The address for CRC calculation

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 70h offset = 21B\_C070h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD0													CRC_ADDR			DATA_END_ADDR								DATA_START_ADDR							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**OCOTP\_CRC\_ADDR field descriptions**

Field	Description
31–19 RSVD0	Reserved
18–16 CRC_ADDR	Address of 32-bit CRC result for comparing
15–8 DATA_END_ADDR	Start address of fuse location for CRC calculation
7–0 DATA_START_ADDR	End address of fuse location for CRC calculation

### 34.5.9 OTP Controller CRC Value Register (OCOTP\_CRC\_VALUE)

The OCOTP Data Register is used for OTP Read

The crc32 value based on CRC\_ADDR

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 80h offset = 21B\_C080h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**OCOTP\_CRC\_VALUE field descriptions**

Field	Description
31–0 DATA	The crc32 value based on CRC_ADDR

**34.5.10 OTP Controller Timing Register (OCOTP\_UMC\_TIMING)**

The OCOTP Data Register is used for OTP Programming

This register specifies timing parameters for programming UMC OCOTP fuse array.

**EXAMPLE**

Empty Example.

Address: 21B\_C000h base + 90h offset = 21B\_C090h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSRVD0																STROBE_PROG_INT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0

**OCOTP\_UMC\_TIMING field descriptions**

Field	Description
31–12 RSRVD0	These bits always read back zero.
11–0 STROBE_PROG_INT	This count value specifies the strobe pulse interval in one time write OTP. $T_{pi} = (STROBE\_PROG + 1) / ipg\_clk\_freq$ . It is given in number of ipg_clk periods.

**34.5.11 OTP Controller Version Register (OCOTP\_VERSION)**

This register always returns a known read value for debug purposes it indicates the version of the block.

This register indicates the RTL version in use.

**EXAMPLE**

Empty Example.



Address: 21B\_C000h base + A0h offset = 21B\_C0A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJOR								MINOR								STEP															
W																																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### OCOTP\_VERSION field descriptions

Field	Description
31–24 MAJOR	Fixed read-only value reflecting the MAJOR field of the RTL version.
23–16 MINOR	Fixed read-only value reflecting the MINOR field of the RTL version.
15–0 STEP	Fixed read-only value reflecting the stepping of the RTL version.

## 34.5.12 Value of OTP Bank0 Word0 (Lock controls) (OCOTP\_LOCK)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 0, word 0 (ADDR = 0x00).

### EXAMPLE

Empty Example.

## OCOTP Memory Map/Register Definition

Address: 21B\_C000h base + 400h offset = 21B\_C400h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	UNALLOCATED		GP_HI_LOCK		GP_LO_LOCK		PIN	RSVD2	-	MISC_CONF	CRC_GP_LOCK		ANALOG		DCP	SW_GP
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD1	SRK	GP2		GP1		MAC_ADDR		RSVD0	SJC_RESP	MEM_TRIM		BOOT_CFG		TESTER	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### OCOTP\_LOCK field descriptions

Field	Description
31–30 UNALLOCATED	Value of un-used portion of LOCK word
29–28 GP_HI_LOCK	Status of shadow register and OTP write lock for GP region. When bit 1 is set, the writing of this region's shadow register is blocked. When bit 0 is set, the writing of this region's OTP fuse word is blocked.
27–26 GP_LO_LOCK	Status of shadow register and OTP write lock for GP region. When bit 1 is set, the writing of this region's shadow register is blocked. When bit 0 is set, the writing of this region's OTP fuse word is blocked.

Table continues on the next page...

**OCOTP\_LOCK field descriptions (continued)**

<b>Field</b>	<b>Description</b>
25 PIN	Status of Pin access lock bit. When set, pin access is disabled.
24 RSVD2	Reserved
23 -	Reserved
22 MISC_CONF	Status of shadow register and OTP write lock for misc_conf region. When set, the writing of this region's shadow register and OTP fuse word are blocked.
21–20 CRC_GP_LOCK	Status of shadow register write and read, OTP program and read lock for CRC region. When bit 1 is set, the reading and writing of this region's OTP fuse and reading of shadow register are blocked. When bit 0 is set, the writing of this region's shadow register and OTP fuse are blocked.
19–18 ANALOG	Status of shadow register and OTP write lock for analog region. When bit 1 is set, the writing of this region's shadow register is blocked. When bit 0 is set, the writing of this region's OTP fuse word is blocked.
17 DCP	Status of shadow register read and write, OTP read and write lock for otpmk region. When set, the writing of this region's shadow register and OTP fuse word are blocked. The read of this region's shadow register and OTP fuse word are also blocked.
16 SW_GP	Status of shadow register lock for the region contained in the SW_GP registers. When set, the over-riding (writing) of this region's shadow bits is blocked. These shadow registers are always readable.
15 RSVD1	Reserved
14 SRK	Status of shadow register and OTP write lock for srk region. When set, the writing of this region's shadow register and OTP fuse word are blocked.
13–12 GP2	Status of shadow register and OTP write lock for gp2 region. When bit 1 is set, the writing of this region's shadow register is blocked. When bit 0 is set, the writing of this region's OTP fuse word is blocked.
11–10 GP1	Status of shadow register and OTP write lock for gp2 region. When bit 1 is set, the writing of this region's shadow register is blocked. When bit 0 is set, the writing of this region's OTP fuse word is blocked.
9–8 MAC_ADDR	Status of shadow register and OTP write lock for mac_addr region. When bit 1 is set, the writing of this region's shadow register is blocked. When bit 0 is set, the writing of this region's OTP fuse word is blocked.
7 RSVD0	Reserved
6 SJC_RESP	Status of shadow register read and write, OTP read and write lock for sjc_resp region. When set, the writing of this region's shadow register and OTP fuse word are blocked. The read of this region's shadow register and OTP fuse word are also blocked.
5–4 MEM_TRIM	Status of shadow register and OTP write lock for mem_trim region. When bit 1 is set, the writing of this region's shadow register is blocked. When bit 0 is set, the writing of this region's OTP fuse word is blocked.
3–2 BOOT_CFG	Status of shadow register and OTP write lock for boot_cfg region. When bit 1 is set, the writing of this region's shadow register is blocked. When bit 0 is set, the writing of this region's OTP fuse word is blocked.
1–0 TESTER	Status of shadow register and OTP write lock for tester region. When bit 1 is set, the writing of this region's shadow register is blocked. When bit 0 is set, the writing of this region's OTP fuse word is blocked.

### 34.5.13 Value of OTP Bank0 Word1 (Configuration and Manufacturing Info.) (OCOTP\_CFG0)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 0, word 1 (ADDR = 0x01).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 410h offset = 21B\_C410h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### OCOTP\_CFG0 field descriptions

Field	Description
31–0 BITS	This register contains 32 bits of the Unique ID and SJC_CHALLENGE field. Reflects value of OTP Bank 0, word 1 (ADDR = 0x01). These bits become read-only after the HW_OCOTP_LOCK_TESTER[1] bit is set.

### 34.5.14 Value of OTP Bank0 Word2 (Configuration and Manufacturing Info.) (OCOTP\_CFG1)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

shadowed memory mapped access to OTP Bank 0, word 2 (ADDR = 0x02).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 420h offset = 21B\_C420h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### OCOTP\_CFG1 field descriptions

Field	Description
31–0 BITS	This register contains 32 bits of the Unique ID and SJC_CHALLENGE field. Reflects value of OTP Bank 0, word 2 (ADDR = 0x02). These bits become read-only after the HW_OCOTP_LOCK_TESTER[1] bit is set.

### 34.5.15 Value of OTP Bank0 Word3 (Configuration and Manufacturing Info.) (OCOTP\_CFG2)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 0, word 3 (ADDR = 0x03).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 430h offset = 21B\_C430h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_CFG2 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 0, word 3 (ADDR = 0x03). These bits become read-only after the HW_OCOTP_LOCK_TESTER[1] bit is set.

### 34.5.16 Value of OTP Bank0 Word4 (Configuration and Manufacturing Info.) (OCOTP\_CFG3)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Non-shadowed memory mapped access to OTP Bank 0, word 4 (ADDR = 0x04).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 440h offset = 21B\_C440h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_CFG3 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 0, word 4 (ADDR = 0x04). These bits become read-only after the HW_OCOTP_LOCK_TESTER[1] bit is set.

### 34.5.17 Value of OTP Bank0 Word5 (Configuration and Manufacturing Info.) (OCOTP\_CFG4)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 0, word 5 (ADDR = 0x05).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 450h offset = 21B\_C450h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_CFG4 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 0, word 5 (ADDR = 0x05). These bits become read-only after the HW_OCOTP_LOCK_BOOT_CFG[1] bit is set.

### 34.5.18 Value of OTP Bank0 Word6 (Configuration and Manufacturing Info.) (OCOTP\_CFG5)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 0, word 6 (ADDR = 0x06).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 460h offset = 21B\_C460h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_CFG5 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 0, word 6 (ADDR = 0x06). These bits become read-only after the HW_OCOTP_LOCK_BOOT_CFG[1] bit is set.

### 34.5.19 Value of OTP Bank0 Word7 (Configuration and Manufacturing Info.) (OCOTP\_CFG6)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 0, word 7 (ADDR = 0x07).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 470h offset = 21B\_C470h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### OCOTP\_CFG6 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 0, word 7 (ADDR = 0x07). These bits become read-only after the HW_OCOTP_LOCK_BOOT_CFG[1] bit is set.

### 34.5.20 Value of OTP Bank1 Word0 (Memory Related Info.) (OCOTP\_MEM0)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP bank 1, word 0 (ADDR = 0x08).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 480h offset = 21B\_C480h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### OCOTP\_MEM0 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP bank 1, word 0 (ADDR = 0x08). These bits become read-only after the HW_OCOTP_LOCK_MEM_TRIM[1] bit is set.

### 34.5.21 Value of OTP Bank1 Word1 (Memory Related Info.) (OCOTP\_MEM1)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP bank 1, word 1 (ADDR = 0x09).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 490h offset = 21B\_C490h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_MEM1 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP bank 1, word 1 (ADDR = 0x09). These bits become read-only after the HW_OCOTP_LOCK_MEM_TRIM[1] bit is set.

### 34.5.22 Value of OTP Bank1 Word2 (Memory Related Info.) (OCOTP\_MEM2)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP bank 1, word 2 (ADDR = 0x0A).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 4A0h offset = 21B\_C4A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	<div>BITS</div>																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_MEM2 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP bank 1, word 2 (ADDR = 0x0A). These bits become read-only after the HW_OCOTP_LOCK_MEM_TRIM[1] bit is set.



### 34.5.23 Value of OTP Bank1 Word3 (Memory Related Info.) (OCOTP\_MEM3)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP bank 1, word 3 (ADDR = 0x0B).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 4B0h offset = 21B\_C4B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_MEM3 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP bank 1, word 3 (ADDR = 0x0B). These bits become read-only after the HW_OCOTP_LOCK_MEM_TRIM[1] bit is set.

### 34.5.24 Value of OTP Bank1 Word4 (Memory Related Info.) (OCOTP\_MEM4)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP bank 1, word 4 (ADDR = 0x0C).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 4C0h offset = 21B\_C4C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_MEM4 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP bank 1, word 4 (ADDR = 0x0C). These bits become read-only after the HW_OCOTP_LOCK_MEM_TRIM[1] bit is set.

### 34.5.25 Value of OTP Bank1 Word5 (Memory Related Info.) (OCOTP\_ANA0)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP bank 1, word 5 (ADDR = 0x0D).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 4D0h offset = 21B\_C4D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_ANA0 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP bank 1, word 5 (ADDR = 0x0D). These bits become read-only after the HW_OCOTP_LOCK_ANALOG[1] bit is set.

### 34.5.26 Value of OTP Bank1 Word6 (General Purpose Customer Defined Info.) (OCOTP\_ANA1)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP bank 1, word 6 (ADDR = 0x0E).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 4E0h offset = 21B\_C4E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_ANA1 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP bank 1, word 6 (ADDR = 0x0E). These bits become read-only after the HW_OCOTP_LOCK_ANALOG[1] bit is set.

### 34.5.27 Value of OTP Bank1 Word7 (General Purpose Customer Defined Info.) (OCOTP\_ANA2)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP bank 1, word 7 (ADDR = 0x0F).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 4F0h offset = 21B\_C4F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BITS																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### OCOTP\_ANA2 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP bank 1, word 7 (ADDR = 0x0F). These bits become read-only after the HW_OCOTP_LOCK_ANALOG[1] bit is set.

### 34.5.28 Shadow Register for OTP Bank2 Word0 (DCP and CRYPTO Key) (OCOTP\_DCP0)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS].

Shadowed memory mapped access to OTP Bank 2, word 0 (ADDR = 0x10).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 500h offset = 21B\_C500h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BITS																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**OCOTP\_DCP0 field descriptions**

Field	Description
31–0 BITS	Shadow register for the DCP Key word0 (Copy of OTP Bank 2, word 0 (ADDR = 0x10)). These bits can be not read and wrotten after the HW_OCOTP_LOCK_DCP bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

**34.5.29 Shadow Register for OTP Bank2 Word1 (DCP and CRYPTO Key) (OCOTP\_DCP1)**

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS].

Shadowed memory mapped access to OTP Bank 2, word 1 (ADDR = 0x11).

**EXAMPLE**

Empty Example.

Address: 21B\_C000h base + 510h offset = 21B\_C510h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**OCOTP\_DCP1 field descriptions**

Field	Description
31–0 BITS	Shadow register for the DCP Key word1 (Copy of OTP Bank 2, word 1 (ADDR = 0x11)). These bits can be not read and wrotten after the HW_OCOTP_LOCK_DCP bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

**34.5.30 Shadow Register for OTP Bank2 Word2 (DCP and CRYPTO Key) (OCOTP\_DCP2)**

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS].

Shadowed memory mapped access to OTP Bank 2, word 2 (ADDR = 0x12).

**EXAMPLE**

Empty Example.

Address: 21B\_C000h base + 520h offset = 21B\_C520h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### OCOTP\_DCP2 field descriptions

Field	Description
31–0 BITS	Shadow register for the DCP Key word2 (Copy of OTP Bank 2, word 2 (ADDR = 0x12)). These bits can be not read and wrotten after the HW_OCOTP_LOCK_DCP bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

## 34.5.31 Shadow Register for OTP Bank2 Word3 (DCP and CRYPTO Key) (OCOTP\_DCP3)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS].

Shadowed memory mapped access to OTP Bank 2, word 3 (ADDR = 0x13).

### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 530h offset = 21B\_C530h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### OCOTP\_DCP3 field descriptions

Field	Description
31–0 BITS	Shadow register for the DCP Key word3 (Copy of OTP Bank 2, word 3 (ADDR = 0x13)). These bits can be not read and wrotten after the HW_OCOTP_LOCK_DCP bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

## 34.5.32 Shadow Register for OTP Bank2 Word4 (DCP Key) (OCOTP\_DCP4)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS].

Shadowed memory mapped access to OTP Bank 2, word 4 (ADDR = 0x14).

### EXAMPLE

## OCOTP Memory Map/Register Definition

Empty Example.

Address: 21B\_C000h base + 540h offset = 21B\_C540h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### OCOTP\_DCP4 field descriptions

Field	Description
31–0 BITS	Shadow register for the DCP Key word4 (Copy of OTP Bank 2, word 4 (ADDR = 0x14)). These bits can be not read and written after the HW_OCOTP_LOCK_DCP bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

## 34.5.33 Shadow Register for OTP Bank2 Word5 (DCP Key) (OCOTP\_DCP5)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS].

Shadowed memory mapped access to OTP Bank 2, word 5 (ADDR = 0x15).

### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 550h offset = 21B\_C550h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### OCOTP\_DCP5 field descriptions

Field	Description
31–0 BITS	Shadow register for the DCP Key word5 (Copy of OTP Bank 2, word 5 (ADDR = 0x15)). These bits can be not read and written after the HW_OCOTP_LOCK_DCP bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

## 34.5.34 Shadow Register for OTP Bank2 Word6 (DCP Key) (OCOTP\_DCP6)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS].

Shadowed memory mapped access to OTP Bank 2, word 6 (ADDR = 0x16).

**EXAMPLE**

Empty Example.

Address: 21B\_C000h base + 560h offset = 21B\_C560h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**OCOTP\_DCP6 field descriptions**

Field	Description
31–0 BITS	Shadow register for the DCP Key word6 (Copy of OTP Bank 2, word 6 (ADDR = 0x16)). These bits can be not read and wrotten after the HW_OCOTP_LOCK_DCP bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

**34.5.35 Shadow Register for OTP Bank2 Word7 (DCP Key) (OCOTP\_DCP7)**

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS].

Shadowed memory mapped access to OTP Bank 2, word 7 (ADDR = 0x17).

**EXAMPLE**

Empty Example.

Address: 21B\_C000h base + 570h offset = 21B\_C570h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**OCOTP\_DCP7 field descriptions**

Field	Description
31–0 BITS	Shadow register for the DCP Key word7 (Copy of OTP Bank 2, word 7 (ADDR = 0x17)). These bits can be not read and wrotten after the HW_OCOTP_LOCK_DCP bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

**34.5.36 Shadow Register for OTP Bank3 Word0 (SRK Hash) (OCOTP\_SRK0)**

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS].

Shadowed memory mapped access to OTP Bank 3, word 0 (ADDR = 0x18).

## EXAMPLE

Empty Example.

Address: 21B\_C000h base + 580h offset = 21B\_C580h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BITS																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## OCOTP\_SRK0 field descriptions

Field	Description
31–0 BITS	Shadow register for the hash of the Super Root Key word0 (Copy of OTP Bank 3, word 0 (ADDR = 0x1C)). These bits become read-only after the HW_OCOTP_LOCK_SRK bit is set.

## 34.5.37 Shadow Register for OTP Bank3 Word1 (SRK Hash) (OCOTP\_SRK1)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS].

Shadowed memory mapped access to OTP Bank 3, word 1 (ADDR = 0x19).

## EXAMPLE

Empty Example.

Address: 21B\_C000h base + 590h offset = 21B\_C590h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BITS																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## OCOTP\_SRK1 field descriptions

Field	Description
31–0 BITS	Shadow register for the hash of the Super Root Key word1 (Copy of OTP Bank 3, word 1 (ADDR = 0x1D)). These bits become read-only after the HW_OCOTP_LOCK_SRK bit is set.

## 34.5.38 Shadow Register for OTP Bank3 Word2 (SRK Hash) (OCOTP\_SRK2)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS].



Shadowed memory mapped access to OTP Bank 3, word 2 (ADDR = 0x1A).

## EXAMPLE

Empty Example.

Address: 21B\_C000h base + 5A0h offset = 21B\_C5A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	<div>BITS</div>																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### OCOTP\_SRK2 field descriptions

Field	Description
31–0 BITS	Shadow register for the hash of the Super Root Key word2 (Copy of OTP Bank 3, word 2 (ADDR = 0x1E)). These bits become read-only after the HW_OCOTP_LOCK_SRK bit is set.

## 34.5.39 Shadow Register for OTP Bank3 Word3 (SRK Hash) (OCOTP\_SRK3)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS].

Shadowed memory mapped access to OTP Bank 3, word 3 (ADDR = 0x1B).

## EXAMPLE

Empty Example.

Address: 21B\_C000h base + 5B0h offset = 21B\_C5B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BITS																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### OCOTP\_SRK3 field descriptions

Field	Description
31–0 BITS	Shadow register for the hash of the Super Root Key word3 (Copy of OTP Bank 3, word 3 (ADDR = 0x1F)). These bits become read-only after the HW_OCOTP_LOCK_SRK bit is set.

## 34.5.40 Shadow Register for OTP Bank3 Word4 (SRK Hash) (OCOTP\_SRK4)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS].

Shadowed memory mapped access to OTP Bank 3, word 4 (ADDR = 0x1C).

## EXAMPLE

Empty Example.

Address: 21B\_C000h base + 5C0h offset = 21B\_C5C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### OCOTP\_SRK4 field descriptions

Field	Description
31–0 BITS	Shadow register for the hash of the Super Root Key word4 (Copy of OTP Bank 3, word 4 (ADDR = 0x20)). These bits become read-only after the HW_OCOTP_LOCK_SRK bit is set.

## 34.5.41 Shadow Register for OTP Bank3 Word5 (SRK Hash) (OCOTP\_SRK5)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS].

Shadowed memory mapped access to OTP Bank 3, word 5 (ADDR = 0x1D).

## EXAMPLE

Empty Example.

Address: 21B\_C000h base + 5D0h offset = 21B\_C5D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### OCOTP\_SRK5 field descriptions

Field	Description
31–0 BITS	Shadow register for the hash of the Super Root Key word5 (Copy of OTP Bank 3, word 5 (ADDR = 0x21)). These bits become read-only after the HW_OCOTP_LOCK_SRK bit is set.

## 34.5.42 Shadow Register for OTP Bank3 Word6 (SRK Hash) (OCOTP\_SRK6)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS].

Shadowed memory mapped access to OTP Bank 3, word 6 (ADDR = 0x1E).

### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 5E0h offset = 21B\_C5E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	<div>BITS</div>																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_SRK6 field descriptions

Field	Description
31–0 BITS	Shadow register for the hash of the Super Root Key word6 (Copy of OTP Bank 3, word 6 (ADDR = 0x22)). These bits become read-only after the HW_OCOTP_LOCK_SRK bit is set.

### 34.5.43 Shadow Register for OTP Bank3 Word7 (SRK Hash) (OCOTP\_SRK7)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS].

Shadowed memory mapped access to OTP Bank 3, word 7 (ADDR = 0x1F).

### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 5F0h offset = 21B\_C5F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_SRK7 field descriptions

Field	Description
31–0 BITS	Shadow register for the hash of the Super Root Key word7 (Copy of OTP Bank 3, word 7 (ADDR = 0x23)). These bits become read-only after the HW_OCOTP_LOCK_SRK bit is set.

### 34.5.44 Value of OTP Bank4 Word0 (Secure JTAG Response Field) (OCOTP\_SJC\_RESP0)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 4, word 0 (ADDR = 0x20).

## EXAMPLE

Empty Example.

Address: 21B\_C000h base + 600h offset = 21B\_C600h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BITS																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### OCOTP\_SJC\_RESP0 field descriptions

Field	Description
31–0 BITS	Shadow register for the SJC_RESP Key word0 (Copy of OTP Bank 4, word 0 (ADDR = 0x20)). These bits can be not read and wrotten after the HW_OCOTP_LOCK_SJC_RESP bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

## 34.5.45 Value of OTP Bank4 Word1 (Secure JTAG Response Field) (OCOTP\_SJC\_RESP1)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 4, word 1 (ADDR = 0x21).

## EXAMPLE

Empty Example.

Address: 21B\_C000h base + 610h offset = 21B\_C610h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BITS																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### OCOTP\_SJC\_RESP1 field descriptions

Field	Description
31–0 BITS	Shadow register for the SJC_RESP Key word1 (Copy of OTP Bank 4, word 1 (ADDR = 0x21)). These bits can be not read and wrotten after the HW_OCOTP_LOCK_SJC_RESP bit is set. If read, returns 0xBADA_BADA and sets HW_OCOTP_CTRL[ERROR].

### 34.5.46 Value of OTP Bank4 Word2 (MAC Address) (OCOTP\_MAC0)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 4, word 2 (ADDR = 0x22).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 620h offset = 21B\_C620h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### OCOTP\_MAC0 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 4, word 2 (ADDR = 0x22).

### 34.5.47 Value of OTP Bank4 Word3 (MAC Address) (OCOTP\_MAC1)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 4, word 3 (ADDR = 0x23).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 630h offset = 21B\_C630h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_MAC1 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 4, word 3 (ADDR = 0x23).

### 34.5.48 Value of OTP Bank4 Word4 (HW Capabilities) (OCOTP\_CRC0)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 4, word 4 (ADDR = 0x24).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 640h offset = 21B\_C640h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_CRC0 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 4, word 4 (ADDR = 0x24).

### 34.5.49 Value of OTP Bank4 Word5 (HW Capabilities) (OCOTP\_CRC1)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 4, word 5 (ADDR = 0x25).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 650h offset = 21B\_C650h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_CRC1 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 4, word 5 (ADDR = 0x25).

### 34.5.50 Value of OTP Bank4 Word6 (HW Capabilities) (OCOTP\_GP1)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 4, word 6 (ADDR = 0x26).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 660h offset = 21B\_C660h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### OCOTP\_GP1 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 4, word 6 (ADDR = 0x26).

### 34.5.51 Value of OTP Bank4 Word7 (HW Capabilities) (OCOTP\_GP2)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 4, word 7 (ADDR = 0x27).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 670h offset = 21B\_C670h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### OCOTP\_GP2 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 4, word 7 (ADDR = 0x27).

### 34.5.52 Value of OTP Bank5 Word0 (HW Capabilities) (OCOTP\_SW\_GP0)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 5, word 0 (ADDR = 0x28).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 680h offset = 21B\_C680h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_SW\_GP0 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 5, word 0 (ADDR = 0x28).

### 34.5.53 Value of OTP Bank5 Word1 (HW Capabilities) (OCOTP\_SW\_GP1)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 5, word 1 (ADDR = 0x29).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 690h offset = 21B\_C690h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_SW\_GP1 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 5, word 1 (ADDR = 0x29).



### 34.5.54 Value of OTP Bank5 Word2 (HW Capabilities) (OCOTP\_SW\_GP2)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 5, word 2 (ADDR = 0x2a).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 6A0h offset = 21B\_C6A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_SW\_GP2 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 5, word 2 (ADDR = 0x2a).

### 34.5.55 Value of OTP Bank5 Word3 (HW Capabilities) (OCOTP\_SW\_GP3)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 5, word 3 (ADDR = 0x2b).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 6B0h offset = 21B\_C6B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_SW\_GP3 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 5, word 3 (ADDR = 0x2b).

### 34.5.56 Value of OTP Bank5 Word4 (HW Capabilities) (OCOTP\_SW\_GP4)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 5, word 4 (ADDR = 0x2c).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 6C0h offset = 21B\_C6C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_SW\_GP4 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 5, word 4 (ADDR = 0x2c).

### 34.5.57 Value of OTP Bank5 Word5 (HW Capabilities) (OCOTP\_MISC\_CONF)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 5, word 5 (ADDR = 0x2d).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 6D0h offset = 21B\_C6D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_MISC\_CONF field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 5, word 5 (ADDR = 0x2d).

### 34.5.58 Value of OTP Bank5 Word6 (HW Capabilities) (OCOTP\_FIELD\_RETURN)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 5, word 6 (ADDR = 0x2e).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 6E0h offset = 21B\_C6E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	<div>BITS</div>																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### OCOTP\_FIELD\_RETURN field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 5, word 6 (ADDR = 0x2e).

### 34.5.59 Value of OTP Bank5 Word7 (HW Capabilities) (OCOTP\_SRK\_REVOKE)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 5, word 7 (ADDR = 0x2f).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 6F0h offset = 21B\_C6F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_SRK\_REVOKE field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 5, word 7 (ADDR = 0x2f).

### 34.5.60 Value of OTP Bank6 Word0 (HW Capabilities) (OCOTP\_GP\_LO0)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 6, word 0 (ADDR = 0x30).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 700h offset = 21B\_C700h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_GP\_LO0 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 6, word 0 (ADDR = 0x30).

### 34.5.61 Value of OTP Bank6 Word1 (HW Capabilities) (OCOTP\_GP\_LO1)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 6, word 1 (ADDR = 0x31).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 710h offset = 21B\_C710h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_GP\_LO1 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 6, word 1 (ADDR = 0x31).

### 34.5.62 Value of OTP Bank6 Word2 (HW Capabilities) (OCOTP\_GP\_LO2)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 6, word 2 (ADDR = 0x32).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 720h offset = 21B\_C720h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_GP\_LO2 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 6, word 2 (ADDR = 0x32).

### 34.5.63 Value of OTP Bank6 Word3 (HW Capabilities) (OCOTP\_GP\_LO3)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 6, word 3 (ADDR = 0x33).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 730h offset = 21B\_C730h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_GP\_LO3 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 6, word 3 (ADDR = 0x33).

### 34.5.64 Value of OTP Bank6 Word4 (HW Capabilities) (OCOTP\_GP\_LO4)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 6, word 4 (ADDR = 0x34).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 740h offset = 21B\_C740h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_GP\_LO4 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 6, word 4 (ADDR = 0x34).

### 34.5.65 Value of OTP Bank6 Word5 (HW Capabilities) (OCOTP\_GP\_LO5)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 6, word 5 (ADDR = 0x35).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 750h offset = 21B\_C750h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_GP\_LO5 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 6, word 5 (ADDR = 0x35).

### 34.5.66 Value of OTP Bank6 Word6 (HW Capabilities) (OCOTP\_GP\_LO6)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 6, word 6 (ADDR = 0x36).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 760h offset = 21B\_C760h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_GP\_LO6 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 6, word 6 (ADDR = 0x36).

### 34.5.67 Value of OTP Bank6 Word7 (HW Capabilities) (OCOTP\_GP\_LO7)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 6, word 7 (ADDR = 0x37).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 770h offset = 21B\_C770h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_GP\_LO7 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 6, word 7 (ADDR = 0x37).

### 34.5.68 Value of OTP Bank7 Word0 (HW Capabilities) (OCOTP\_GP\_HI0)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 7, word 0 (ADDR = 0x38).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 780h offset = 21B\_C780h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_GP\_HI0 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 7, word 0 (ADDR = 0x38).

### 34.5.69 Value of OTP Bank7 Word1 (HW Capabilities) (OCOTP\_GP\_HI1)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 7, word 1 (ADDR = 0x39).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 790h offset = 21B\_C790h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_GP\_HI1 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 7, word 1 (ADDR = 0x39).



### 34.5.70 Value of OTP Bank7 Word2 (HW Capabilities) (OCOTP\_GP\_HI2)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 7, word 2 (ADDR = 0x3a).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 7A0h offset = 21B\_C7A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### OCOTP\_GP\_HI2 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 7, word 2 (ADDR = 0x3a).

### 34.5.71 Value of OTP Bank7 Word3 (HW Capabilities) (OCOTP\_GP\_HI3)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 7, word 3 (ADDR = 0x3b).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 7B0h offset = 21B\_C7B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_GP\_HI3 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 7, word 3 (ADDR = 0x3b).

### 34.5.72 Value of OTP Bank7 Word4 (HW Capabilities) (OCOTP\_GP\_HI4)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 7, word 4 (ADDR = 0x3c).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 7C0h offset = 21B\_C7C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### OCOTP\_GP\_HI4 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 7, word 4 (ADDR = 0x3c).

### 34.5.73 Value of OTP Bank7 Word5 (HW Capabilities) (OCOTP\_GP\_HI5)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 7, word 5 (ADDR = 0x3d).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 7D0h offset = 21B\_C7D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	BITS																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### OCOTP\_GP\_HI5 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 7, word 5 (ADDR = 0x3d).

### 34.5.74 Value of OTP Bank7 Word6 (HW Capabilities) (OCOTP\_GP\_HI6)

Copied from the OTP automatically after reset. Can be re-loaded by setting OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 7, word 6 (ADDR = 0x3e).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 7E0h offset = 21B\_C7E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BITS																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### OCOTP\_GP\_HI6 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 7, word 6 (ADDR = 0x3e).

### 34.5.75 Value of OTP Bank7 Word7 (HW Capabilities) (OCOTP\_GP\_HI7)

Copied from the OTP automatically after reset. Can be re-loaded by setting HW\_OCOTP\_CTRL[RELOAD\_SHADOWS]

Shadowed memory mapped access to OTP Bank 7, word 7 (ADDR = 0x3f).

#### EXAMPLE

Empty Example.

Address: 21B\_C000h base + 7F0h offset = 21B\_C7F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BITS																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### OCOTP\_GP\_HI7 field descriptions

Field	Description
31–0 BITS	Reflects value of OTP Bank 7, word 7 (ADDR = 0x3f).

**OCOTP\_GP\_HI7 field descriptions (continued)**

Field	Description
-------	-------------

## Chapter 35

# On-Chip RAM Memory Controller (OCRAM)

### 35.1 Overview

There are 2 OCRAM controllers implemented in i.MX 6SoloLite. One controller is for the normal 128KB on-chip RAM. The other controller is for the 256KB L2 cache of ARM platform.

The L2 cache can be configured into "OCRAM mode" and used in the same way as normal OCRAM. The on-chip RAM block is implemented as a slave module on the 64-bit system AXI bus. Designed as a simple on-chip memory controller, it supports only one AXI port with memory banks. For the AXI port, the read and write transactions are handled by two independent modules. As it is possible to have simultaneous read and write request from the AXI bus, each memory bank has an arbiter with round-robin scheme. After arbitration, the granted read or write access command can then be issued to the memory cell through a read/write MUX.

The 4 memory banks are organized with lower 2 bits of address which is AXI bus address and is 64 bits aligned interleaved. This allows a read access and a write access can be processed at the same time if they are targeted to different memory banks.

Various options are provided for adding pipeline or wait-states in read/write access, in order to ensure flexible timing control at both high and low frequencies.

The internal block diagram is shown in the figure below.

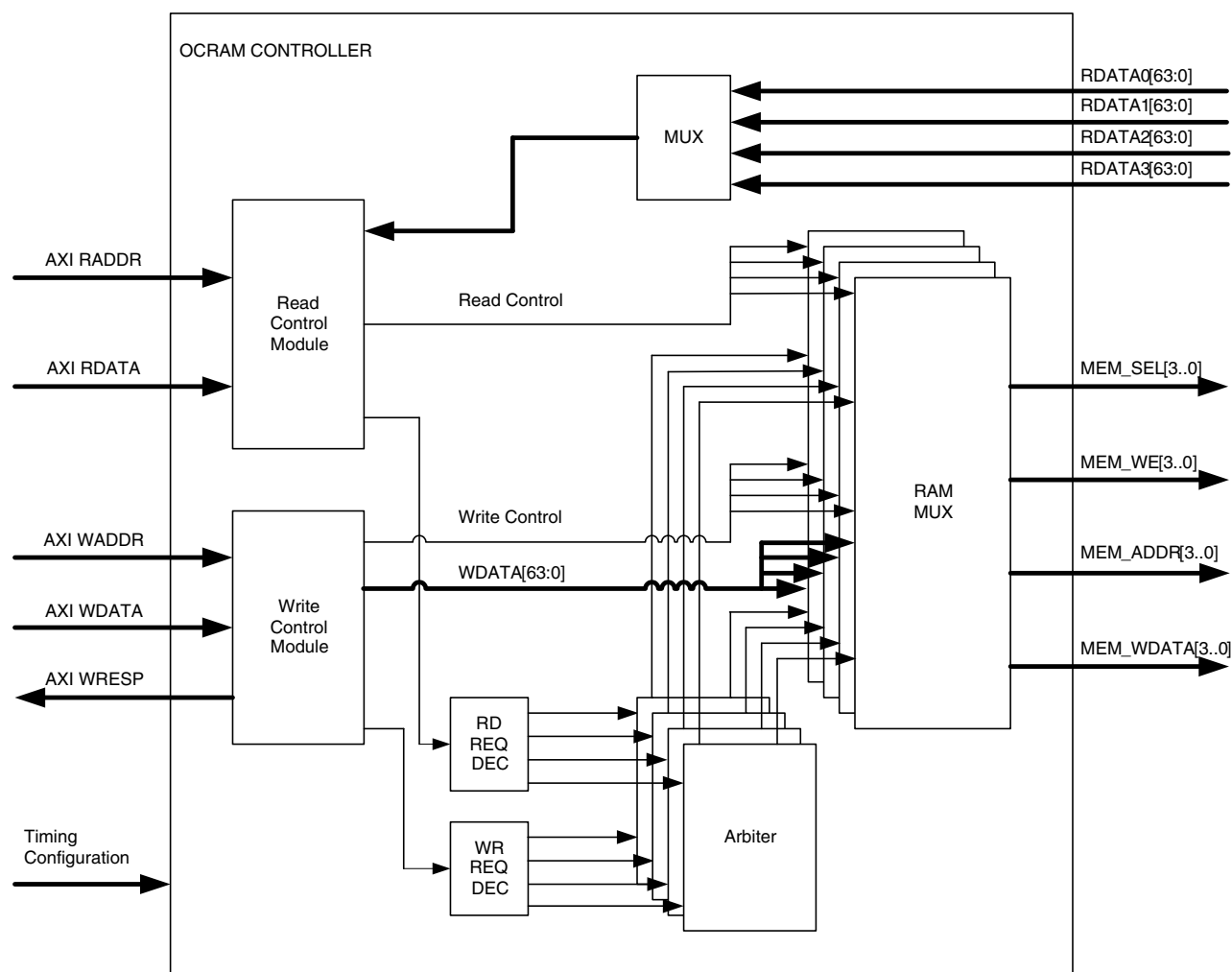


Figure 35-1. On-chip RAM Block Diagram

## 35.2 Basic Functions

### 35.2.1 Read/Write Arbitration

The arbiter is used to handle the read/write request from the read control module and write control module. It grants read/write access to the memory cell according to the internal arbitration state and the request/switch signal from the read control module and write control module. Generally, the arbitration is using the round-robin method.

The detailed rules used in arbitration are as follows:

- If there is no granted read or write in the last cycle, and there is only a read request or a write request, the request will be granted.
- If there is no granted read or write in the last cycle, and there are both read or write requests coming in at the same time, the read request will be granted first.
- If a granted read/write transaction has just finished, the write/read request will have the higher priority in the next cycle.
- If the first read/write access request in a transaction is granted, all the data transfer in this burst will be finished before the next arbitration begins, that is, the round-robin arbitration mechanism is based on AXI transaction, not data access.

### 35.2.2 TrustZone

TrustZone is also supported on this block.

When SECURE\_ENBL bit in the General Purpose Register (IOMUXC\_GPR10) bits [10:4] and [26:20] is set, the STARTADDR and ENDADDR bit-fields in this register establish the region of OCRAM that can only be accessed (both read and write) according to the execution mode policy described in CSU chapter, [Peripheral access policy](#). If this bit is cleared to zero, the entire OCRAM can be accessed in either secure or non-secure mode. The TrustZone bits shows in [Programmable Registers](#).

#### NOTE

The ENDADDR is not configurable and its value is the last address of the OCRAM space. The STARTADDR granularity is of 4KB.

## 35.3 Advanced Features

This section describes some advanced features designed to avoid timing issues when the on-chip RAM is working at high frequency.

All of the features can be disabled/enabled by programming the corresponding fields of the General Purpose Register (IOMUXC.GPR3) bits [24:21] and bits [3:0] in the IOMUX chapter.

### 35.3.1 Read Data Wait State

When the wait state is enabled, it will cost 2 cycles for each read access, (each beat of a read burst).

This can avoid the potential timing problem caused by the relatively longer memory access time at higher frequency.

When this feature is disabled, it only costs 1 clock cycle to finish a read transaction, that is, to get read data back in the next cycle of read request becomes valid on the bus.

For the normal ocram, the read data wait state is configurable via IOMUXC.GPR3[21]. For the L2 cache as ocram, the read data wait state is configurable via IOMUXC.GPR3[0].

### **35.3.2 Read Address Pipeline**

When this feature is enabled, the read address from the AXI master is delayed 1 cycle before it can be accepted by the on-chip RAM.

This can avoid setup time issues for the read access on the memory cell at high frequency. Enabling this feature can cost, at most, 1 more clock cycle for each AXI read transaction, that is, at most 1 more clock cycle for each read burst with multiple beats of data.

When this feature is disabled, the read address from the AXI master can be accepted by the on-chip RAM without delay, and data can become ready for master at next clock cycle (if no other access and no read data wait).

For the normal ocram, the read address pipeline is configurable via IOMUXC.GPR3[22]. For the L2 cache as ocram, the read address pipeline is configurable via IOMUXC.GPR3[1].

### **35.3.3 Write Data Pipeline**

When this feature is enabled, the write data from the AXI master would be delayed 1 cycle before it can be accepted by the on-chip RAM.

This can avoid setup time issue for the write access on the memory cell at high frequency. Enabling this feature would cost at most 1 more clock cycle for each AXI write transaction, that is, at most 1 more clock cycle for each write burst with multiple beats of data.

When this feature is disabled, the write data from the AXI master can be accepted by the on-chip RAM without delay, and data can be written to memory at this cycle (if no other access and write address is also ready at this cycle).



For the normal ocram, the write data pipeline is configurable via IOMUXC.GPR3[23].  
For the L2 cache as ocram, the write data pipeline is configurable via IOMUXC.GPR3[2].

### 35.3.4 Write Address Pipeline

When this feature is enabled, the write address from the AXI master would be delayed 1 cycle before it can be accepted by the on-chip RAM.

This can avoid setup time issue for the write access on the memory cell at high frequency. Enabling this feature would cost at most 1 more clock cycle for each AXI write transaction, that is, at most 1 more clock cycle for each write burst with multiple beats of data.

When this feature is disabled, the write address from the AXI master can be accepted by the on-chip RAM without delay, and data can be written to memory at this cycle (if no other access and write data is also ready at this cycle).

For the normal ocram, the write address pipeline is configurable via IOMUXC.GPR3[24].  
For the L2 cache as ocram, the write address pipeline is configurable via IOMUXC.GPR3[3]

## 35.4 Programmable Registers

There are no programmable registers in this block; however, OCRAM configurable bits can be found in the IOMUX Controller (IOMUXC) general purpose registers found here.

- TrustZone bits: IOMUXC\_GPR10
- WAIT state / Pipeline bits: IOMUXC\_GPR3
- L2 Cache OCRAM enable bits: IOMUXC\_GPR11

(See [IOMUXC Memory Map/Register Definition](#) for more details).



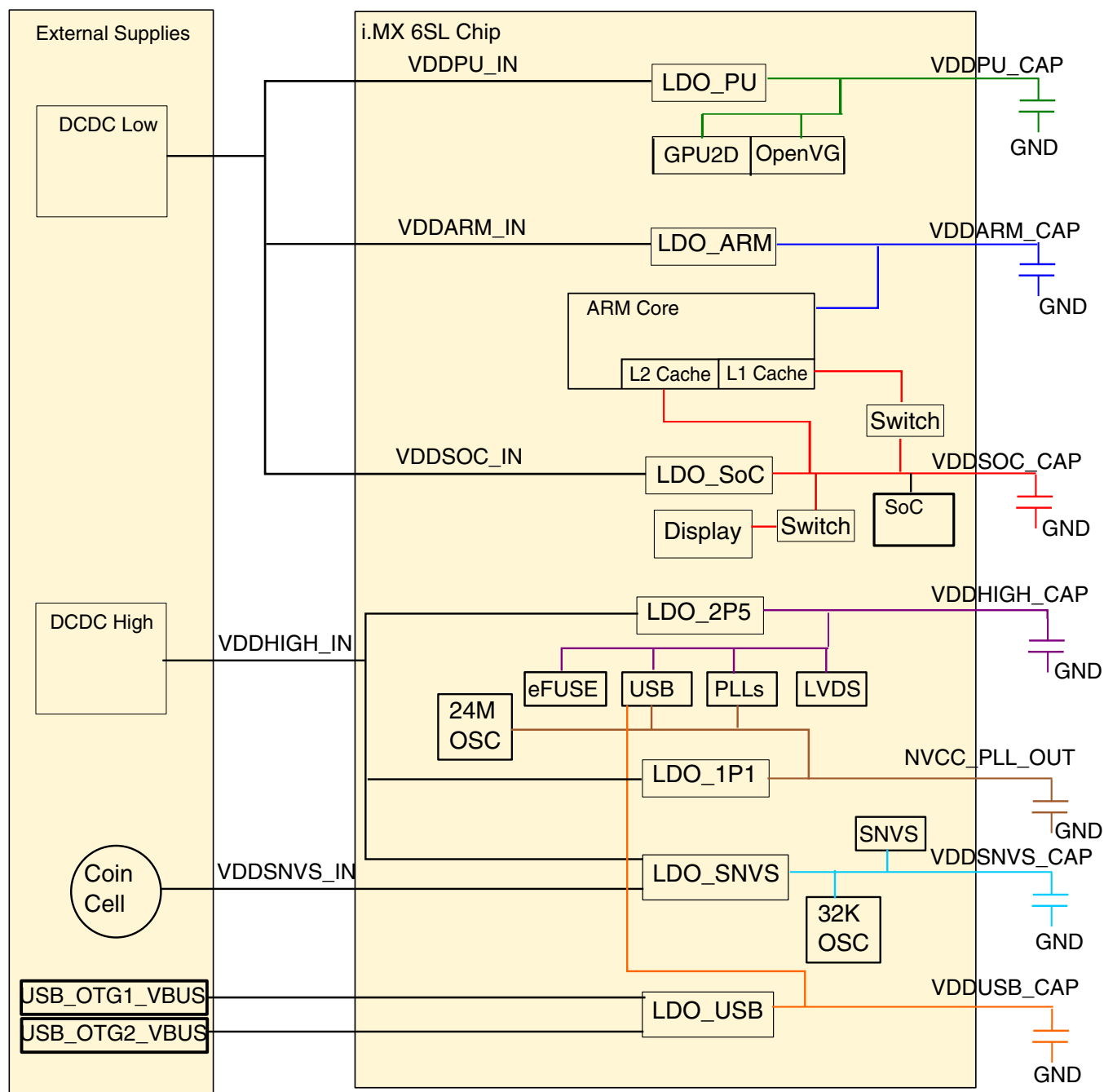
## **Chapter 36**

# **Power Management Unit (PMU)**

### **36.1 Overview**

The power management unit (PMU) is designed to simplify the external power interface. The power system can be split into the input power sources and their characteristics, the integrated power transforming and controlling elements, and the final load interconnection and requirements.

A typical power system utilizing the PMU is depicted below.



**Figure 36-1. Power system overview**

Utilizing seven LDO regulators, the number of external supplies is greatly reduced. Not counting the backup coin and USB inputs, the number of external supplies is reduced to two. Missing from this external supply total is the number of necessary external supplies to power the desired memory interface; that number varies depending on the type of

external memory selected. Other supplies may also be necessary to supply the voltage to the different I/O power segments if their I/O voltages have to be different from what is provided above.

## 36.2 Digital LDO Regulators

The PMU has three digital LDO regulators. They are referred to as "digital" because of the logic loads they drive, not because of their construction. These regulators have three basic modes that are unique to the digital regulators.

- **Internal Bypass**—The regulation FET is switched fully on passing the external input voltage to the load unaltered. The analog part of the regulator is powered down in this state, removing any loss other than the IR drop through the power grid and the FET. (TARG = 0x1F)
- **External Bypass**—The input and output of the regulator are shorted externally to the part. If operating in this configuration, enable the internal bypass early in the start-up sequence before attempting high-frequency/high-power operation.
- **Power Gate**—The regulation FET is switched off fully, limiting the current draw from the supply. The analog part of the regulator is powered down, limiting the power consumption. The output voltage falls to a level at which the residual leakage of the power FET balances with the leakage of the load. (TARG = 0x00)
- **Analog regulation mode**—The regulation FET is controlled such that the output voltage of the regulator equals the programmed target voltage. The target voltage is fully programmable in 25-mV steps.

These modes allow the regulators to implement voltage scaling and power gating and allow bypass. With the bypass feature, all of the accuracy and control requirements can be shifted to the external supply source if capable and desired.

These digital regulators also feature brownout detection which is helpful when supplies are starting to collapse. The voltage value where brownout is signaled is programmable as an offset from the programmed target voltage. The controls are located in the PMU\_MISC2 register. The core is interrupted on a brownout.

The three digital regulators are known as LDO\_ARM (#0), LDO\_PU (#1), and LDO\_SOC (#2). As shown in the power system overview figure, the ARM regulator powers the ARM cores. The PU powers the GPU and display portions of the chip. The SOC regulator powers the rest of the digital logic on the chip. The target voltages of these regulators are all reset to 1.1 V on startup. All regulators support generous programming ranges in 25-mV steps. It is possible to program voltages above the process limit for the

chip, thus causing permanent damage. Likewise, it is possible to program the voltage so low that the chip cannot continue to operate or even retain state without clocks. Care should be taken with these settings.

Care must be taken when raising the output voltage of the regulator rapidly. This can cause large currents to flow into the output cap of the regulator up to the limits of the input supply. When the input supply capability is exceeded, this can cause an input supply dip that may affect other regulators on the same supply. Therefore, the rate of voltage change on the output of the regulator should be limited. When powering up the regulator, the integrated current limiter controls the ramp rate. This limiter is only effective when transitioning from the off state of the regulator (bypassed or power gated). However, in a DVFS situation, the same high rate of change can occur if the target voltage is raised rapidly by software. To limit the rate of change, the hardware controlling the regulator effects a piecewise linear ramp by stepping the output voltage in 25-mV steps until the desired output voltage is reached. The slope of the ramp is controlled by the time spent at each 25-mV step and is controlled by the step time field in the PMU\_MISC2 register. The same situation is not a problem when the output voltage is dropped as the load pulls down the output cap. As a result, any reduction in the programmed regulator target voltage is immediately effective with the actual supply voltage falling at a rate controlled by the load on the regulator.

## 36.3 Analog LDO Regulators

There are two analog regulators described here.

### 36.3.1 LDO 1P1

The LDO\_1P1 module on the chip implements a programmable linear-regulator function from a higher analog supply voltage (2.8 V–3.3 V) to produce a nominal 1.1-V output voltage.

The output of the regulator can be programmed in 25-mV steps from 0.8 V to 1.4 V and can provide up to 150 mA output current. The regulator has been designed to be stable with a minimum external low-ESR decoupling capacitor of 2.2  $\mu$ F, though the actual capacitance required should be determined by the application. A programmable brownout detector is included in the regulator which can be used by the system to determine when the load capability of the regulator is being exceeded, so the necessary steps can be taken. Current limiting can be enabled by setting the PMU\_REG\_1P1[ENABLE\_ILIMIT] bit to

allow for in-rush current requirements during startup if needed. Active pulldown can also be enabled by setting the PMU\_REG\_1P1[ENABLE\_PULLDOWN] bit for systems requiring this feature.

### 36.3.2 LDO 2P5

The LDO\_2P5 module on the chip implements a programmable linear-regulator function from a higher analog supply voltage (2.8V-3.3V) to produce a nominal 2.5V output voltage.

The output of the regulator can be programmed in 25mV steps from 2.0V to 2.75V and can provide up to 350mA output current with a 300mV drop-out voltage. The regulator has been designed to be stable with a minimum external low-ESR decoupling capacitor of 2.2 $\mu$ F, though the actual capacitance required should be determined by the application. A programmable brown-out detector is included in the regulator which can be used by the system to determine when the load capability of the regulator is being exceeded to take the necessary steps. Current-limiting can be enabled by setting the REG\_PMU\_2P5[ENABLE\_ILIMIT] bit to allow for in-rush current requirements during start-up if needed. Active-pulldown can also be enabled by setting the REG\_PMU\_2P5[ENABLE\_PULLDOWN] bit for systems requiring this feature.

#### 36.3.2.1 Low Power Operation

The 2.5 V LDO includes an alternate, self-biased, low-precision, weak regulator which can be enabled for applications needing to keep the 2.5-V output voltage alive during low-power modes where the main regulator and its associated global bandgap reference module are disabled. The output of this weak regulator is not programmable and is a function of its input power supply as well as load current. Typically, with a 3-V input power supply, the weak regulator output is 2.525 V, and its output impedance is approximately 40  $\Omega$ .

The low-power mode is enabled by setting high the PMU\_REG\_2P5[ENABLE\_WEAK\_LINREG] bit of the regulator. It is recommended that the following sequence be followed to enable this mode:

1. Throttle down the 2.5-V attached load to its low-power maintain state.
2. Disable the main 2.5-V regulator driver by clearing the PMU\_REG\_2P5[ENABLE\_LINREG] bit.
3. Enable the weak 2.5-V regulator by setting the PMU\_REG\_2P5[ENABLE\_WEAK\_LINREG] bit.

To go back to full-power operation, reverse the steps outlined above. Note that the external decoupling cap is supporting the power supply between steps 2 and 3. Therefore step 3 should happen appropriately in time relative to the discharge of the supporting capacitor.

## 36.4 USB LDO Regulator

The USB\_LDO module on the chip implements a programmable linear-regulator function from the USB VBUS voltages (typically 5 V) to produce a nominal 3.0-V output voltage.

The output of the regulator can be programmed in 25-mV steps, from 2.625V to 3.4 V, and can provide up to 50mA output current. The regulator has been designed to be stable with a minimum external low-ESR decoupling capacitor of 2.2  $\mu$ F, though the actual capacitance required should be determined by the application. A programmable brownout detector is included in the regulator which can be used by the system to determine when the load capability of the regulator is being exceeded, so the necessary steps can be taken. This regulator has a built-in power mux which allows the user to choose to run the regulator from either VBUS supply when both are present. If only one of the VBUS voltages is present, then the regulator automatically selects this supply. Current limit is also included to help the system meet in-rush current targets.

Upon attachment of VBUS, this regulator starts up in a low-power, self-preservation mode to prevent over-voltage conditions on the chip. It is expected that the user transition to full regulation by enabling the regulator and disabling the in-rush current limits via its control registers. Upon VBUS removal, it is further expected that the regulator controls are returned to their reset state.

## 36.5 SNVS Regulator

The SNVS regulator takes the SNVS\_IN supply and generates the SNVS\_CAP supply, which powers the real time clock and SNVS blocks.

If VDDHIGH\_IN is present, then the SNVS\_IN supply is internally shorted to the VDDHIGH\_IN supply to allow coin cell recharging if necessary. The output voltage is regulated to a 1.1V level.

VDDHIGH\_IN can be separated from SNVS\_IN by setting the `discon_high_snvs` bit in the `PMU_misc0`.



## 36.6 Power Modes

### 36.6.1 Reverse Well Biasing

The reverse well biasing module on the chip includes a self-clocked/self-regulating charge-pump circuit to generate a negative bias voltage for the floating PWELL, and a low-power regulator to generate a positive bias voltage for the NWELL of digital logic cells on the SOC power domain.

Static leakage reduction can be achieved through the use of these reverse well bias voltages. Typical power consumption of the module is 50  $\mu$ A when driving a 10-nF purely capacitive load.

## 36.7 PMU Memory Map/Register Definition

The register definitions that affect the behavior of the digital LDO regulators follow.

### NOTE

Some of the registers are collections of bits that affect multiple components on the chip. Those that are not pertinent to this chapter have comments in the related register bitfields.

If a full description is desired, please consult the full register programming reference in the related block.

**PMU memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20C_8140	Digital Regulator Core Register (PMU_REG_CORE)	32	R/W	0040_2010h	<a href="#">36.7.1/2246</a>
20C_8150	Miscellaneous Register 0 (PMU_MISC0)	32	R/W	0400_0000h	<a href="#">36.7.2/2249</a>
20C_8160	Miscellaneous Register 1 (PMU_MISC1)	32	R/W	0000_0000h	<a href="#">36.7.3/2252</a>
20C_8164	Miscellaneous Register 1 (PMU_MISC1_SET)	32	R/W	0000_0000h	<a href="#">36.7.3/2252</a>
20C_8168	Miscellaneous Register 1 (PMU_MISC1_CLR)	32	R/W	0000_0000h	<a href="#">36.7.3/2252</a>
20C_816C	Miscellaneous Register 1 (PMU_MISC1_TOG)	32	R/W	0000_0000h	<a href="#">36.7.3/2252</a>
20C_8170	Miscellaneous Control Register (PMU_MISC2)	32	R/W	0027_2727h	<a href="#">36.7.4/2255</a>
20C_8174	Miscellaneous Control Register (PMU_MISC2_SET)	32	R/W	0027_2727h	<a href="#">36.7.4/2255</a>
20C_8178	Miscellaneous Control Register (PMU_MISC2_CLR)	32	R/W	0027_2727h	<a href="#">36.7.4/2255</a>
20C_817C	Miscellaneous Control Register (PMU_MISC2_TOG)	32	R/W	0027_2727h	<a href="#">36.7.4/2255</a>

## 36.7.1 Digital Regulator Core Register (PMU\_REG\_CORE)

This register defines the function of the digital regulators

Address: 20C\_8000h base + 140h offset = 20C\_8140h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		FET_ODRIVE	Reserved		REG2_ADJ				REG2_TARG				REG1_ADJ		
W																
Reset	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	REG1_ADJ		REG1_TARG				REG0_ADJ				REG0_TARG					
W																
Reset	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0

### PMU\_REG\_CORE field descriptions

Field	Description
31–30 -	This field is reserved.
29 FET_ODRIVE	If set, increases the gate drive on power gating FETs to reduce leakage in the off state. Care must be taken to apply this bit only when the input supply voltage to the power FET is less than 1.1V.  <b>NOTE:</b> This bit should only be used in low-power modes where the external input supply voltage is nominally 0.9V.
28–27 -	This field is reserved.
26–23 REG2_ADJ	This field defines the adjustment bits to calibrate the target value of REG2 (REG_SOC). The adjustment is applied on top of any adjustment applied to the global reference in the MISC0 register.  0000 No adjustment 0001 + 0.25% 0010 + 0.50% 0011 + 0.75% 0100 + 1.00% 0101 + 1.25% 0110 + 1.50% 0111 + 1.75% 1000 - 0.25% 1001 - 0.50%

Table continues on the next page...

**PMU\_REG\_CORE field descriptions (continued)**

Field	Description
	1010 - 0.75% 1011 - 1.00% 1100 - 1.25% 1101 - 1.50% 1110 - 1.75% 1111 - 2.00%
22–18 REG2_TARG	This field defines the target voltage for the SOC power domain. Single-bit increments reflect 25mV core voltage steps. Some steps may not be relevant because of input supply limitations or load operation.  00000 Power gated off 00001 Target core voltage = 0.725V 10000 Target core voltage = 1.100V 11110 Target core voltage = 1.450V 11111 Power FET switched full on. No regulation.
17–14 REG1_ADJ	This field defines the adjustment bits to calibrate the target value of REG1 (REG_PU). The adjustment is applied on top of any adjustment applied to the global reference in the MISC0 register.  0000 No adjustment 0001 + 0.25% 0010 + 0.50% 0011 + 0.75% 0100 + 1.00% 0101 + 1.25% 0110 + 1.50% 0111 + 1.75% 1000 - 0.25% 1001 - 0.50% 1010 - 0.75% 1011 - 1.00% 1100 - 1.25% 1101 - 1.50% 1110 - 1.75% 1111 - 2.00%
13–9 REG1_TARG	This field defines the target voltage for the VPU/GPU power domain. Single-bit increments reflect 25mV core voltage steps. Some steps may not be relevant because of input supply limitations or load operation.  00000 Power gated off 00001 Target core voltage = 0.725V 10000 Target core voltage = 1.100V 11110 Target core voltage = 1.450V 11111 Power FET switched full on. No regulation.
8–5 REG0_ADJ	This field defines the adjustment bits to calibrate the target value of REG1 (ARM_CORE). The adjustment is applied on top of any adjustment applied to the global reference in the MISC0 register.  0000 No adjustment 0001 + 0.25% 0010 + 0.50% 0011 + 0.75%

*Table continues on the next page...*

## PMU\_REG\_CORE field descriptions (continued)

Field	Description
	0100 + 1.00% 0101 + 1.25% 0110 + 1.50% 0111 + 1.75% 1000 - 0.25% 1001 - 0.50% 1010 - 0.75% 1011 - 1.00% 1100 - 1.25% 1101 - 1.50% 1110 - 1.75% 1111 - 2.00%
4-0 REG0_TARG	<p>This field defines the target voltage for the arm core power domain. Single-bit increments reflect 25mV core voltage steps. Some steps may not be relevant because of input supply limitations or load operation.</p> 00000 Power gated off 00001 Target core voltage = 0.725V 10000 Target core voltage = 1.100V 11110 Target core voltage = 1.450V 11111 Power FET switched full on. No regulation.

## 36.7.2 Miscellaneous Register 0 (PMU\_MISC0)

This register defines the control and status bits for miscellaneous analog blocks.

Address: 20C\_8000h base + 150h offset = 20C\_8150h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved			CLKGATE_DELAY			CLKGATE_CTRL	Reserved				WBCP_VPW_THRESH			OSC_XTALOK_EN	OSC_XTALOK
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	OSC_I		discon_high_snvs	STOP_MODE_CONFIG		Reserved			REFTOP_VBGUP	REFTOP_VBGADJ			REFTOP_SELFBIASOFF	Reserved		REFTOP_PWD
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## PMU\_MISC0 field descriptions

Field	Description
31–29 -	This field is reserved.
28–26 CLKGATE_ DELAY	<p>This field specifies the delay between powering up the XTAL 24MHz clock and releasing the clock to the digital logic inside the analog block.</p> <p><b>NOTE:</b> Do not change the field during a low power event. This is not a field that the user would normally need to modify.</p> <p><b>NOTE:</b> Not related to PMU.</p> <p>000 0.5ms 001 1.0ms 010 2.0ms 011 3.0ms 100 4.0ms 101 5.0ms 110 6.0ms 111 7.0ms</p>
25 CLKGATE_CTRL	<p>This bit allows disabling the clock gate (always ungated) for the xtal 24MHz clock that clocks the digital logic in the analog block.</p> <p><b>NOTE:</b> Do not change the field during a low power event. This is not a field that the user would normally need to modify.</p> <p><b>NOTE:</b> Not related to PMU.</p> <p>0 <b>ALLOW_AUTO_GATE</b> — Allow the logic to automatically gate the clock when the XTAL is powered down. 1 <b>NO_AUTO_GATE</b> — Prevent the logic from ever gating off the clock.</p>
24–20 -	This field is reserved. Always set to zero.
19–18 WBCP_VPW_ THRESH	<p>This signal alters the voltage that the pwell is charged pumped to.</p> <p>00 <b>NOMINAL_BIAS</b> — Nominal output pwell bias voltage. 01 <b>PLUS_25MV</b> — Increase pwell output voltage by 25mV. 10 <b>MINUS_25MV</b> — Decrease pwell output pwell voltage by 25mV. 11 <b>MINUS_50MV</b> — Decrease pwell output pwell voltage by 50mV.</p>
17 OSC_XTALOK_ EN	<p>This bit enables the detector that signals when the 24MHz crystal oscillator is stable.</p> <p><b>NOTE:</b> Not related to PMU, Clocking content</p>
16 OSC_XTALOK	<p>Status bit that signals that the output of the 24-MHz crystal oscillator is stable. Generated from a timer and active detection of the actual frequency.</p> <p><b>NOTE:</b> Not related to PMU, clocking content.</p>
15–14 OSC_I	<p>This field determines the bias current in the 24MHz oscillator. The aim is to start up with the highest bias current, which can be decreased after startup if it is determined to be acceptable.</p> <p><b>NOTE:</b> Not related to PMU.</p>

*Table continues on the next page...*

## PMU\_MISC0 field descriptions (continued)

Field	Description
	00 <b>NOMINAL</b> — Nominal 01 <b>MINUS_12_5_PERCENT</b> — Decrease current by 12.5% 10 <b>MINUS_25_PERCENT</b> — Decrease current by 25.0% 11 <b>MINUS_37_5_PERCENT</b> — Decrease current by 37.5%
13 discon_high_ snvs	This bit forces the short between VDDHIGH_IN and VSNVS_IN to open when asserted. This is useful in power cases where SNVS_IN > VDDHIGH_IN.
12–11 STOP_MODE_ CONFIG	Configure the analog behavior in stop mode.  0x0 <b>DEEP</b> — Deep Stop Mode - All analog except rtc powered down on stop mode assertion 0x1 <b>LIGHT</b> — Light Stop Mode - All the analog domain except the LDO_1P1, LDO_2P5, and PLL3 are powered down on STOP mode assertion. If required the CCM can be configured not to power down the oscillator (XTALOSC). PLL3 can be disabled with register settings if desired.  0x2 — Reserved 0x3 — Reserved
10–8 -	This field is reserved. Reserved
7 REFTOP_ VBGUP	Status bit that signals the analog bandgap voltage is up and stable. 1 - Stable.
6–4 REFTOP_ VBGADJ	000 Nominal VBG 001 VBG+0.78% 010 VBG+1.56% 011 VBG+2.34% 100 VBG-0.78% 101 VBG-1.56% 110 VBG-2.34% 111 VBG-3.12%
3 REFTOP_ SELFBIASOFF	Control bit to disable the self-bias circuit in the analog bandgap. The self-bias circuit is used by the bandgap during startup. This bit should be set after the bandgap has stabilized and is necessary for best noise performance of analog blocks using the outputs of the bandgap.  <b>NOTE:</b> Value should be returned to zero before removing vddhigh_in or asserting bit 0 of this register (REFTOP_PWD) to assure proper restart of the circuit.  0 Uses coarse bias currents for startup 1 Uses bandgap-based bias currents for best performance.
2–1 -	This field is reserved.
0 REFTOP_PWD	Control bit to power-down the analog bandgap reference circuitry.  <b>NOTE:</b> A note of caution, the bandgap is necessary for correct operation of most of the LDO, pll, and other analog functions on the die.

### 36.7.3 Miscellaneous Register 1 (PMU\_MISC1n)

This register defines the control and status bits for miscellaneous analog blocks. The LVDS1 and LVDS2 controls below control the behavior of the anack1/1b LVDS IO.

Address: 20C\_8000h base + 160h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	IRQ_DIG_BO	IRQ_ANA_BO	IRQ_TEMPSENSE	Reserved												
W	w1c	w1c	w1c													
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		Reserved	LVDSCLK1_IBEN	Reserved	LVDSCLK1_OBEN	Reserved						LVDS1_CLK_SEL			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



## PMU\_MISC1n field descriptions

Field	Description
31 IRQ_DIG_BO	This status bit is set to one when when any of the digital regulator brownout interrupts assert. Check the regulator status bits to discover which regulator interrupt asserted.
30 IRQ_ANA_BO	This status bit is set to one when when any of the analog regulator brownout interrupts assert. Check the regulator status bits to discover which regulator interrupt asserted.
29 IRQ_TEMPSENSE	This status bit is set to one when when the temperature sensor interrupt asserts. <b>NOTE:</b> Not related to PMU, Temperature Monitor content.
28–14 -	This field is reserved.
13 -	This field is reserved. Reserved
12 LVDSCLK1_IBEN	This enables the LVDS input buffer for anack1/1b. Do not enable input and output buffers simultaneously. <b>NOTE:</b> Not related to PMU, Clocking content.
11 -	This field is reserved. Reserved
10 LVDSCLK1_OBEN	This enables the LVDS output buffer for anack1/1b. Do not enable input and output buffers simultaneously. <b>NOTE:</b> Not related to PMU, clocking content.
9–5 -	This field is reserved. Reserved
4–0 LVDS1_CLK_SEL	This field selects the clk to be routed to anack2/2b. <b>NOTE:</b> Not related to PMU.  <div> 00000      <b>ARM_PLL</b> — Arm PLL  00001      <b>SYS_PLL</b> — System PLL  00010      <b>PFD4</b> — pfd4  00011      <b>PFD5</b> — pfd5  00100      <b>PFD6</b> — pfd6  00101      <b>PFD7</b> — pfd7  00110      <b>AUDIO_PLL</b> — Audio PLL  00111      <b>VIDEO_PLL</b> — Video PLL  01000      Not functional  01001      <b>ETHERNET_REF</b> — ethernet ref clock  01010      Not Functional  01011      Not Functional  01100      <b>USB1_PLL</b> — USB1 PLL clock  01101      <b>USB2_PLL</b> — USB2 PLL clock  01110      <b>PFD0</b> — pfd0  01111      <b>PFD1</b> — pfd1  10000      <b>PFD2</b> — pfd2  10001      <b>PFD3</b> — pfd3  10010      <b>XTAL</b> — xtal </div>

Table continues on the next page...

**PMU\_MISC1*n* field descriptions (continued)**

Field	Description
10011	Not Functional
10100	Not Functional
10101 to 11111	pfd7

### 36.7.4 Miscellaneous Control Register (PMU\_MISC2n)

This register defines the control for miscellaneous PMU Analog blocks.

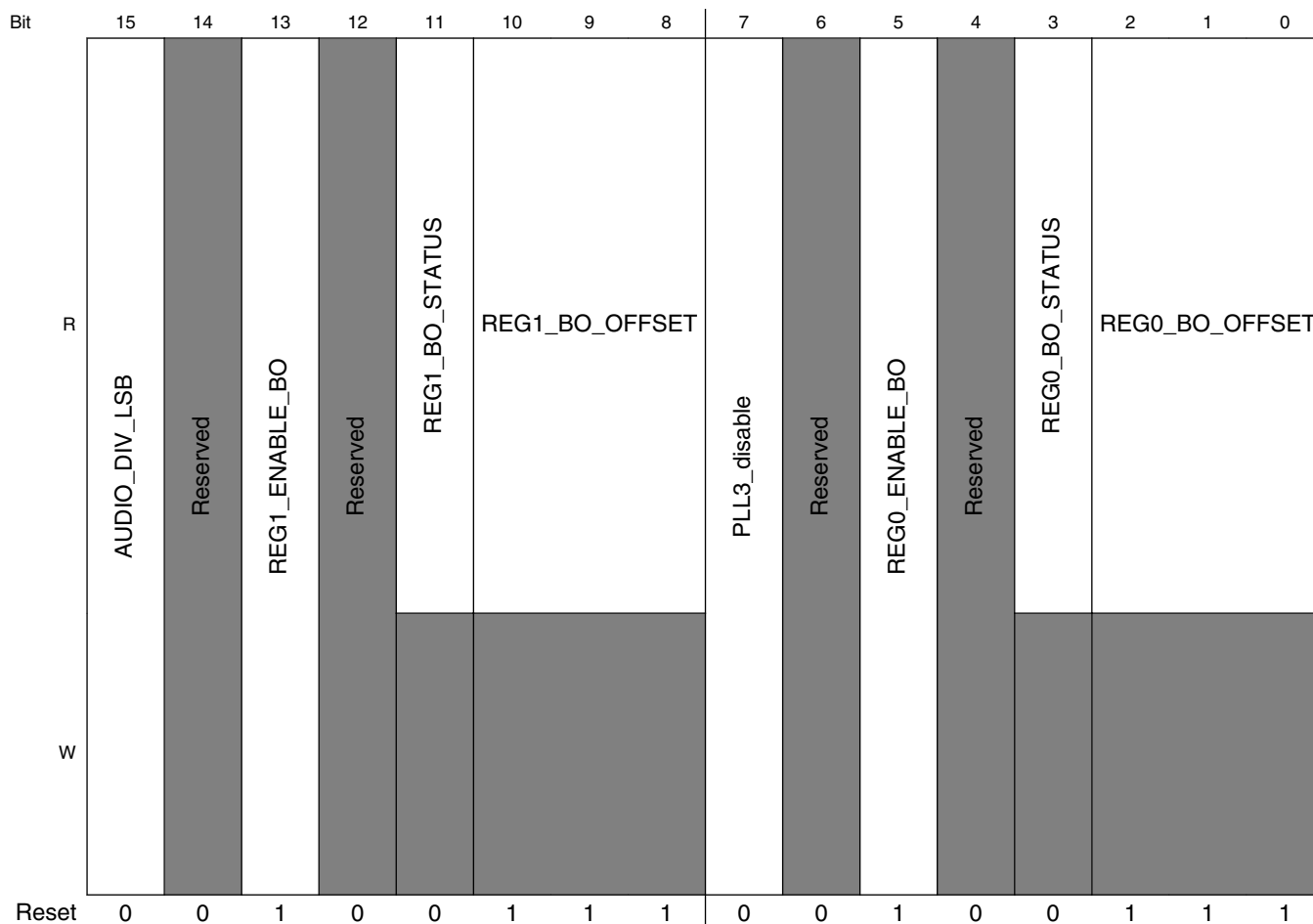
#### NOTE

This register is shared with CCM.

Address: 20C\_8000h base + 170h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1
	VIDEO_DIV		REG2_STEP_TIME		REG1_STEP_TIME		REG0_STEP_TIME		AUDIO_DIV_MSB	REG2_OK	REG2_ENABLE_BO	Reserved	REG2_BO_STATUS	REG2_BO_OFFSET		

## PMU Memory Map/Register Definition



### PMU\_MISC2n field descriptions

Field	Description
31–30 VIDEO_DIV	<p>Post-divider for video. The output clock of the video PLL should be gated prior to changing this divider to prevent glitches. This divider is feed by PLL_VIDEOOn[POST_DIV_SELECT] to achieve division ratios of / 1, /2, /4, /8, and /16.</p> <p><b>NOTE:</b> Not related to PMU. See <a href="#">Clock Controller Module (CCM)</a> for more information.</p> <p>00 divide by 1 (Default)  01 divide by 2  10 divide by 1  11 divide by 4</p>
29–28 REG2_STEP_TIME	<p>Number of clock periods (24MHz clock).</p> <p>00 <b>64_CLOCKS</b> — 64  01 <b>128_CLOCKS</b> — 128  10 <b>256_CLOCKS</b> — 256  11 <b>512_CLOCKS</b> — 512</p>
27–26 REG1_STEP_TIME	<p>Number of clock periods (24MHz clock).</p> <p>00 <b>64_CLOCKS</b> — 64  01 <b>128_CLOCKS</b> — 128</p>

Table continues on the next page...

## PMU\_MISC2n field descriptions (continued)

Field	Description
	10 <b>256_CLOCKS</b> — 256 11 <b>512_CLOCKS</b> — 512
25–24 REG0_STEP_ TIME	Number of clock periods (24MHz clock).  00 <b>64_CLOCKS</b> — 64 01 <b>128_CLOCKS</b> — 128 10 <b>256_CLOCKS</b> — 256 11 <b>512_CLOCKS</b> — 512
23 AUDIO_DIV_ MSB	MSB of Post-divider for Audio PLL. The output clock of the video PLL should be gated prior to changing this divider to prevent glitches. This divider is feed by PLL_AUDION[POST_DIV_SELECT] to achieve division ratios of /1, /2, /4, /8, and /16.  <b>NOTE:</b> MSB bit value pertains to the first bit, please program the LSB bit (bit 15) as well to change divider value  <b>NOTE:</b> Not related to PMU. See <a href="#">Clock Controller Module (CCM)</a> for more information.  00 divide by 1 (Default) 01 divide by 2 10 divide by 1 11 divide by 4
22 REG2_OK	Signals that the voltage is above the brownout level for the SOC supply. 1 = regulator output > brownout_target
21 REG2_ENABLE_ BO	Enables the brownout detection.
20 -	This field is reserved.
19 REG2_BO_ STATUS	Reg2 brownout status bit.
18–16 REG2_BO_ OFFSET	This field defines the brown out voltage offset for the xPU power domain. IRQ_DIG_BO is also asserted. Single-bit increments reflect 25mV brownout voltage steps. The reset brown-offset is 175mV below the programmed target code. Brownout target = OUTPUT_TRG - BO_OFFSET. Some steps may be irrelevant because of input supply limitations or load operation.  100 Brownout offset = 0.100V 111 Brownout offset = 0.175V
15 AUDIO_DIV_LSB	LSB of Post-divider for Audio PLL. The output clock of the video PLL should be gated prior to changing this divider to prevent glitches. This divider is feed by PLL_AUDION[POST_DIV_SELECT] to achieve division ratios of /1, /2, /4, /8, and /16.  <b>NOTE:</b> LSB bit value pertains to the last bit, please program the MSB bit (bit 23) as well, to change divider value  <b>NOTE:</b> Not related to PMU. See <a href="#">Clock Controller Module (CCM)</a> for more information.  00 divide by 1 (Default) 01 divide by 2 10 divide by 1 11 divide by 4

Table continues on the next page...

## PMU\_MISC2n field descriptions (continued)

Field	Description
14 -	This field is reserved. Reserved
13 REG1_ENABLE_ BO	Enables the brownout detection.
12 -	This field is reserved.
11 REG1_BO_ STATUS	Reg1 brownout status bit. 1 Brownout, supply is below target minus brownout offset.
10–8 REG1_BO_ OFFSET	This field defines the brown out voltage offset for the xPU power domain. IRQ_DIG_BO is also asserted. Single-bit increments reflect 25mV brownout voltage steps. The reset brown-offset is 175mV below the programmed target code. Brownout target = OUTPUT_TRG - BO_OFFSET. Some steps may be irrelevant because of input supply limitations or load operation.  100 Brownout offset = 0.100V 111 Brownout offset = 0.175V
7 PLL3_disable	Default value of "0". Should be set to "1" to turn off the USB-PLL(PLL3) in run mode. <b>NOTE:</b> Not related to PMU. See <a href="#">Clock Controller Module (CCM)</a> for more information.
6 -	This field is reserved.
5 REG0_ENABLE_ BO	Enables the brownout detection.
4 -	This field is reserved.
3 REG0_BO_ STATUS	Reg0 brownout status bit. 1 Brownout, supply is below target minus brownout offset.
2–0 REG0_BO_ OFFSET	This field defines the brown out voltage offset for the CORE power domain. IRQ_DIG_BO is also asserted. Single-bit increments reflect 25mV brownout voltage steps. Some steps may be irrelevant because of input supply limitations or load operation.  100 Brownout offset = 0.100V 111 Brownout offset = 0.175V

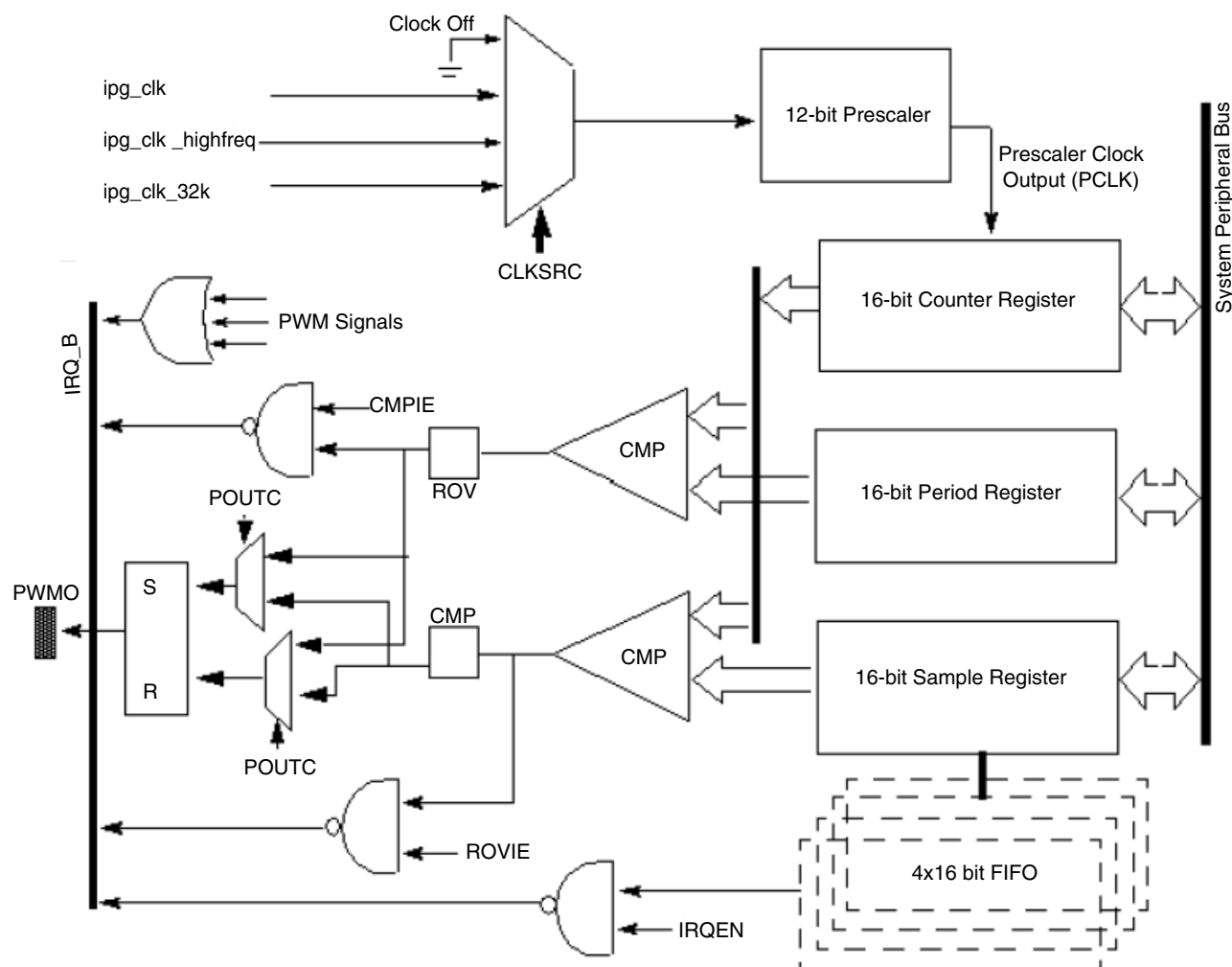
## **Chapter 37**

# **Pulse Width Modulation (PWM)**

### **37.1 Overview**

The Pulse Width Modulation (PWM) has a 16-bit counter, and is optimized to generate sound from stored sample audio images and it can also generate tones. It uses 16-bit resolution and a 4 x 16 data FIFO.

This section presents an overview of the PWM. A block diagram of the PWM module is shown in the figure below.



### Figure 37-1. Pulse-Width Modulator Block Diagram

The following features characterize the PWM:

- 16-bit up-counter with clock source selection
- 4 x 16 FIFO to minimize interrupt overhead
- 12-bit prescaler for division of clock
- Sound and melody generation
- Active high or active low configured output
- Can be programmed to be active in low-power mode
- Can be programmed to be active in debug mode
- Interrupts at compare and rollover

## 37.2 External Signals



The PWM follows IP Bus protocol when interfacing with the processor core. PWM does not have any interface signals with any other block inside the chip except for clock and reset inputs from the Clock Control Module (CCM), System Reset Controller (SRC), and interrupt signals to the processor interrupt handler. There is a single output signal.

The following table outlines the external signals.

**Table 37-1. PWM External Signals**

Signal	Description	Pad	Mode	Direction
PWM1_OUT	This is the PWM1 functional output of the PWM. A modulated signal of the block is observed at this pin. It can be viewed as a clock signal whose period and duty cycle can be varied with different settings of the cycle of 50%.	EPDC_SDCE2	ALT2	O
		LCD_DAT0	ALT3	
		PWM1	ALT0	
		UART1_RXD	ALT1	
		HSIC_DAT	ALT2	
PWM2_OUT	This is the PWM2 functional output of the PWM. A modulated signal of the block is observed at this pin. It can be viewed as a clock signal whose period and duty cycle can be varied with different settings of the cycle of 50%.	EPDC_SDCE3	ALT2	O
		LCD_DAT1	ALT3	
		UART1_TXD	ALT1	
		HSIC_STROBE	ALT2	
PWM3_OUT	This is the PWM3 functional output of the PWM. A modulated signal of the block is observed at this pin. It can be viewed as a clock signal whose period and duty cycle can be varied with different settings of the cycle of 50%.	AUD_TXFS	ALT1	O
		EPDC_SDCE0	ALT2	
		LCD_DAT2	ALT3	
		REF_CLK_24M	ALT2	
PWM4_OUT	This is the PWM4 functional output of the PWM. A modulated signal of the block is observed at this pin. It can be viewed as a clock signal whose period and duty cycle can be varied with different settings of the cycle of 50%.	AUD_MCLK	ALT1	O
		EPDC_GDSP	ALT1	
		EPDC_SDCE1	ALT2	
		FEC_REF_CLK	ALT3	
		LCD_CLK	ALT4	
		LCD_DAT3	ALT3	
		REF_CLK_32K	ALT2	

## 37.3 Clocks

The table found here describes the clock sources for PWM.

Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 37-2. PWM Clocks**

Clock name	Clock Root	Description
ipg_clk	ipg_clk_root	Peripheral clock
ipg_clk_32k	ckil_sync_clk_root	low-frequency reference clock (32kHz)
ipg_clk_highfreq	perclk_clk_root	high-frequency reference clock
ipg_clk_s	ipg_clk_root	Peripheral access clock

The clock that feeds the prescaler can be selected from:

- High-frequency reference clock (ipg\_clk\_highfreq) pat\_ref or CKIH

This is a high frequency clock, provided by the Clock Control Module (CCM). This clock should be on in the low power mode when the ipg\_clk is turned off. Thus, the PWM can be run on this clock in the low power mode.

- Low-frequency reference clock (ipg\_clk\_32k, CKIL)

This is the 32 KHz low reference clock which is provided by the CCM. This clock should be on in the low power mode when ipg\_clk is turned off. Thus, PWM can be run on this clock in the low power mode.

- Peripheral clock (ipg\_clk)

This clock should be on in normal operations. In low power mode, it can be switched off.

- Peripheral access clock (ipg\_clk\_s)

This clock is used for register read/write.

The clock input source is determined by the PWM control register field PWM\_CR[CLKSRC]. The CLKSRC value should only be changed when the PWM is disabled.

A change in the value of the PRESCALER field of the control register is immediately reflected on its output clock frequency.

## 37.4 Functional Description

The following sections detail the PWM operation and function.

### 37.4.1 Operation

The output of the PWM is a toggling signal whose frequency and duty cycle can be modulated by programming the appropriate registers. It has a 16-bit up counter which counts from 0x0000 until the counter value equals the PWM\_PR + 1. After this match occurs the counter is reset to 0x0000.

At the beginning of a count period cycle, the PWM0 pin is set to one (default) and the counter begins counting up from 0x0000. The sample value in the sample FIFO is compared on each count of prescaler clock. When the sample and count values match, the PWM0 signal is cleared to zero (default). The counter continues counting until the period match occurs and subsequently another period cycle begins.

When the PWM is enabled, the counter starts running and generates an output with the reset values in the period and sample registers. It is recommended that the programming of these registers be done before PWM is enabled.

A hardware reset results in all the PWM count and sample registers being cleared and the FIFO being flushed. The control register shows that FIFO is empty and it can be written into, and the PWM is disabled. A software reset has the same results, however the state of the STOPEN, DOZEN, WAITEN, and DBGEN bits in the control register are not affected. Software reset can be asserted even when the PWM is in disabled state.

#### 37.4.1.1 FIFO

Digital sample values can be loaded into the pulse-width modulator as 16-bit words. The endianness can be changed using the BCTR and HCTR bits of the control register. A 4-word (16-bit) FIFO minimizes interrupt overhead. A maskable interrupt is generated when the number of data words fall below the water level set by the FWM field in the control register.

A write to the PWM\_SAR sample register results in the value being stored into the FIFO if it is not full. A write when the FIFO is full sets FWE (FIFO write error) bit in the status register and the FIFO contents remain unchanged. The FIFO can be written at any time, but can be read only when the PWM is enabled. The PWM\_SR[FIFOAV] field shows how many data words are currently contained in the FIFO and whether or not it can be written into.

A read on the sample register yields the current FIFO value that is being used, or will be used, by the PWM for generation on the output signal. Therefore, a write and a subsequent read on the sample register may result in different values being obtained.

### 37.4.1.2 Rollover and Compare Event

The counter is reset to 0x0000 after its value equals the PWM\_PR[PERIOD] + 1 and resumes counting thereafter. This event is referred to as a rollover. For example, if PWM\_PR[PERIOD] = 0x0000, the counter is reset when it equals 0x0001. When PWM\_PR[PERIOD] = 0xFFFF or 0xFFFE, the counter is reset when it equals 0xFFFF. For more information, see the PWM Period Register (PWM\_PR) description.

During a rollover event the output is either set (default), reset or has no effect according to the programming of the POUTC field in the control register. This event can also generate an interrupt if the respective interrupt enable bit is set in the control register.

When the counter value reaches the sample value, the output of the PWM is reset (default), set or has no effect according to the programming of the POUTC field of control register. This event is referred to as a compare event. This event can also generate an interrupt if the respective interrupt enable bit is set in the control register.

If the rollover event sets the PWM output signal, the compare event will reset it and vice versa for a particular programming configuration of POUTC field.

### 37.4.1.3 Low Power Mode Behavior

In low power mode, if the clock from the selected clock source is available, the PWM counter continues to run and an output is produced, depending on whether the control bit for that mode is set or not. In the absence of the clock itself, or if the corresponding low power bit in the control register is 0, the counter is reset and resumes counting when it exits the low power mode.

### 37.4.1.4 Debug Mode Behavior

In debug mode, PWM has the option of continuing to run or be halted. If the DBGEN bit is not set in the PWM\_PWMCR, the PWM is halted. If the DBGEN bit is set, then the PWM will continue to run in the debug mode.

## 37.5 Enable Sequence for the PWM

The sequence found here should be used to enable the PWM.

1. Configure the desired settings for the PWM Control Register (PWMx\_PWMCR) while keeping the PWM disabled (PWMx\_PWMCR[0]=0).
2. Enable the desired interrupts in the PWM Interrupt Register (PWMx\_PWMIR).
3. One to three initial samples may be written to the PWM Sample Register (PWMx\_PWMSAR). The initial sample values will be loaded into the PWM FIFO even if the PWM is not yet enabled. Do not write a 4th sample because the FIFO will become full and trigger a FIFO Write Error (FWE). This error will prevent the PWM from starting once it is enabled.
4. Check the FIFO Write Error status bit (FWE), the Compare status bit (CMP) and the Roll-over status bit (ROV) in the PWM Status Register (PWMx\_PWMSR) to make sure they are all zero. Any non-zero status bits should be cleared by writing a 1 to them.
5. Write the desired period to the PWM Period Register (PWMx\_PWMPR).
6. Enable the PWM by writing a 1 to the PWM Enable bit, PWMx\_PWMCR[0], while maintaining the other register bits in their previously configured state.

## 37.6 Disable Sequence for the PWM

The PWM can be disabled at any time by clearing the PWM enable bit, PWMx\_PWMCR[0] to 0.

Any data remaining in the FIFO will not be produced at the PWM output after the PWM has been disabled and will remain in the FIFO until the PWM is enabled again. A software reset (setting PWMx\_PWMCR[3] to 1) or a hardware reset will clear the FIFO and any remaining data will be lost.

## 37.7 PWM Memory Map/Register Definition

The PWM includes six user-accessible 32-bit registers.

**PWM memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
208_0000	PWM Control Register (PWM1_PWMCR)	32	R/W	0000_0000h	<a href="#">37.7.1/2267</a>
208_0004	PWM Status Register (PWM1_PWMSR)	32	w1c	0000_0008h	<a href="#">37.7.2/2269</a>
208_0008	PWM Interrupt Register (PWM1_PWMIR)	32	R/W	0000_0000h	<a href="#">37.7.3/2270</a>

*Table continues on the next page...*

## PWM memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
208_000C	PWM Sample Register (PWM1_PWMSAR)	32	R/W	0000_0000h	<a href="#">37.7.4/2271</a>
208_0010	PWM Period Register (PWM1_PWMPR)	32	R/W	0000_FFEh	<a href="#">37.7.5/2272</a>
208_0014	PWM Counter Register (PWM1_PWMCNR)	32	R	0000_0000h	<a href="#">37.7.6/2273</a>
208_4000	PWM Control Register (PWM2_PWMCR)	32	R/W	0000_0000h	<a href="#">37.7.1/2267</a>
208_4004	PWM Status Register (PWM2_PWMSR)	32	w1c	0000_0008h	<a href="#">37.7.2/2269</a>
208_4008	PWM Interrupt Register (PWM2_PWMIR)	32	R/W	0000_0000h	<a href="#">37.7.3/2270</a>
208_400C	PWM Sample Register (PWM2_PWMSAR)	32	R/W	0000_0000h	<a href="#">37.7.4/2271</a>
208_4010	PWM Period Register (PWM2_PWMPR)	32	R/W	0000_FFEh	<a href="#">37.7.5/2272</a>
208_4014	PWM Counter Register (PWM2_PWMCNR)	32	R	0000_0000h	<a href="#">37.7.6/2273</a>
208_8000	PWM Control Register (PWM3_PWMCR)	32	R/W	0000_0000h	<a href="#">37.7.1/2267</a>
208_8004	PWM Status Register (PWM3_PWMSR)	32	w1c	0000_0008h	<a href="#">37.7.2/2269</a>
208_8008	PWM Interrupt Register (PWM3_PWMIR)	32	R/W	0000_0000h	<a href="#">37.7.3/2270</a>
208_800C	PWM Sample Register (PWM3_PWMSAR)	32	R/W	0000_0000h	<a href="#">37.7.4/2271</a>
208_8010	PWM Period Register (PWM3_PWMPR)	32	R/W	0000_FFEh	<a href="#">37.7.5/2272</a>
208_8014	PWM Counter Register (PWM3_PWMCNR)	32	R	0000_0000h	<a href="#">37.7.6/2273</a>
208_C000	PWM Control Register (PWM4_PWMCR)	32	R/W	0000_0000h	<a href="#">37.7.1/2267</a>
208_C004	PWM Status Register (PWM4_PWMSR)	32	w1c	0000_0008h	<a href="#">37.7.2/2269</a>
208_C008	PWM Interrupt Register (PWM4_PWMIR)	32	R/W	0000_0000h	<a href="#">37.7.3/2270</a>
208_C00C	PWM Sample Register (PWM4_PWMSAR)	32	R/W	0000_0000h	<a href="#">37.7.4/2271</a>
208_C010	PWM Period Register (PWM4_PWMPR)	32	R/W	0000_FFEh	<a href="#">37.7.5/2272</a>
208_C014	PWM Counter Register (PWM4_PWMCNR)	32	R	0000_0000h	<a href="#">37.7.6/2273</a>

## 37.7.2 PWM Control Register (PWMx\_PWMCR)

The PWM control register (PWM\_PWMCR) is used to configure the operating settings of the PWM. It contains the prescaler for the clock division.

Address: Base address + 0h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0				FWM		STOPEN	DOZEN	WAITEN	DBGEN	BCTR	HCTR	POUTC		CLKSRC	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PRESCALER												SWR	REPEAT		EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### PWMx\_PWMCR field descriptions

Field	Description
31–28 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
27–26 FWM	FIFO Water Mark. These bits are used to set the data level at which the FIFO empty flag will be set and the corresponding interrupt generated  00 FIFO empty flag is set when there are more than or equal to 1 empty slots in FIFO 01 FIFO empty flag is set when there are more than or equal to 2 empty slots in FIFO 10 FIFO empty flag is set when there are more than or equal to 3 empty slots in FIFO 11 FIFO empty flag is set when there are more than or equal to 4 empty slots in FIFO
25 STOPEN	Stop Mode Enable. This bit keeps the PWM functional while in stop mode. When this bit is cleared, the input clock is gated off in stop mode. This bit is not affected by software reset. It is cleared by hardware reset.  0 Inactive in stop mode 1 Active in stop mode
24 DOZEN	Doze Mode Enable. This bit keeps the PWM functional in doze mode. When this bit is cleared, the input clock is gated off in doze mode. This bit is not affected by software reset. It is cleared by hardware reset.  0 Inactive in doze mode 1 Active in doze mode
23 WAITEN	Wait Mode Enable. This bit keeps the PWM functional in wait mode. When this bit is cleared, the input clock is gated off in wait mode. This bit is not affected by software reset. It is cleared by hardware reset.

Table continues on the next page...

**PWMx\_PWMCR field descriptions (continued)**

Field	Description
	0 Inactive in wait mode 1 Active in wait mode
22 DBGEN	Debug Mode Enable. This bit keeps the PWM functional in debug mode. When this bit is cleared, the input clock is gated off in debug mode. This bit is not affected by software reset. It is cleared by hardware reset.  0 Inactive in debug mode 1 Active in debug mode
21 BCTR	Byte Data Swap Control. This bit determines the byte ordering of the 16-bit data when it goes into the FIFO from the sample register.  0 byte ordering remains the same 1 byte ordering is reversed
20 HCTR	Half-word Data Swap Control. This bit determines which half word data from the 32-bit IP Bus interface is written into the lower 16 bits of the sample register.  0 Half word swapping does not take place 1 Half words from write data bus are swapped
19–18 POUTC	PWM Output Configuration. This bit field determines the mode of PWM output on the output pin.  00 Output pin is set at rollover and cleared at comparison 01 Output pin is cleared at rollover and set at comparison 10 PWM output is disconnected 11 PWM output is disconnected
17–16 CLKSRC	Select Clock Source. These bits determine which clock input will be selected for running the counter. After reset the system functional clock is selected. The input clock can also be turned off if these bits are set to 00. This field value should only be changed when the PWM is disabled  00 Clock is off 01 ipg_clk 10 ipg_clk_highfreq 11 ipg_clk_32k
15–4 PRESCALER	Counter Clock Prescaler Value. This bit field determines the value by which the clock will be divided before it goes to the counter.  0x000 Divide by 1 0x001 Divide by 2 0xff Divide by 4096
3 SWR	Software Reset. PWM is reset when this bit is set to 1. It is a self clearing bit. A write 1 to this bit is a single wait state write cycle. When the block is in reset state this bit is set and is cleared when the reset procedure is over. Setting this bit resets all the registers to their reset values except for the STOPEN, DOZEN, WAITEN, and DBGEN bits in this control register.  0 PWM is out of reset 1 PWM is undergoing reset
2–1 REPEAT	Sample Repeat. This bit field determines the number of times each sample from the FIFO is to be used.  00 Use each sample once 01 Use each sample twice 10 Use each sample four times 11 Use each sample eight times

*Table continues on the next page...*



**PWMx\_PWMCR field descriptions (continued)**

Field	Description
0 EN	<p>PWM Enable. This bit enables the PWM. If this bit is not enabled, the clock prescaler and the counter is reset. When the PWM is enabled, it begins a new period, the output pin is set to start a new period while the prescaler and counter are released and counting begins.</p> <p>To make the PWM work with softreset and disable/enable, users can do software reset by setting the SWR bit, wait software reset done, configure the registers, and then enable the PWM by setting this bit to "1"</p> <p>Users can also disable/enable the PWM if PWM would like to be stopped and resumed with same registers configurations .</p> <p>0    PWM disabled 1    PWM enabled</p>

**37.7.3 PWM Status Register (PWMx\_PWMSR)**

The PWM status register (PWM\_PWMSR) contains seven bits which display the state of the FIFO and the occurrence of rollover and compare events. The FIFOAV bit is read-only but the other four bits can be cleared by writing 1 to them. The FE, ROV, and CMP bits are associated with FIFO-Empty, Roll-over, and Compare interrupts, respectively.

Address: Base address + 4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0									FWE	CMP	ROV	FE	FIFOAV		
W										w1c	w1c	w1c	w1c			
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

**PWMx\_PWMSR field descriptions**

Field	Description
31–7 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
6 FWE	<p>FIFO Write Error Status. This bit shows that an attempt has been made to write FIFO when it is full.</p> <p>0    FIFO write error not occurred 1    FIFO write error occurred</p>
5 CMP	<p>Compare Status. This bit shows that a compare event has occurred.</p> <p>0    Compare event not occurred 1    Compare event occurred</p>

*Table continues on the next page...*

**PWMx\_PWMSR field descriptions (continued)**

Field	Description
4 ROV	Roll-over Status. This bit shows that a roll-over event has occurred.  0 Roll-over event not occurred 1 Roll-over event occurred
3 FE	FIFO Empty Status Bit. This bit indicates the FIFO data level in comparison to the water level set by FWM field in the control register.  0 Data level is above water mark 1 When the data level falls below the mark set by FWM field
2–0 FIFOAV	FIFO Available. These read-only bits indicate the data level remaining in the FIFO. An attempted write to these bits will not affect their value and no transfer error is generated.  000 No data available 001 1 word of data in FIFO 010 2 words of data in FIFO 011 3 words of data in FIFO 100 4 words of data in FIFO 101 unused 110 unused 111 unused

**37.7.4 PWM Interrupt Register (PWMx\_PWMIR)**

The PWM Interrupt register (PWM\_PWMIR) contains three bits which control the generation of the compare, rollover and FIFO empty interrupts.

Address: Base address + 8h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														CIE	RIE
W																FIE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**PWMx\_PWMIR field descriptions**

Field	Description
31–3 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
2 CIE	Compare Interrupt Enable. This bit controls the generation of the Compare interrupt.

*Table continues on the next page...*

**PWMx\_PWMIR field descriptions (continued)**

Field	Description
	0 Compare Interrupt not enabled 1 Compare Interrupt enabled
1 RIE	Roll-over Interrupt Enable. This bit controls the generation of the Rollover interrupt.  0 Roll-over interrupt not enabled 1 Roll-over Interrupt enabled
0 FIE	FIFO Empty Interrupt Enable. This bit controls the generation of the FIFO Empty interrupt.  0 FIFO Empty interrupt disabled 1 FIFO Empty interrupt enabled

**37.7.5 PWM Sample Register (PWMx\_PWMSAR)**

The PWM sample register (PWM\_PWMSAR) is the input to the FIFO. 16-bit words are loaded into the FIFO. The FIFO can be written at any time, but can be read only when the PWM is enabled. The PWM will run at the last set duty-cycle setting if all the values of the FIFO has been utilized, until the FIFO is reloaded or the PWM is disabled. When a new value is written, the duty cycle changes after the current period is over.

A value of zero in the sample register will result in the PWMO output signal always being low/high (POUTC = 00 it will be low and POUTC = 01 it will be high), and no output waveform will be produced. If the value in this register is higher than the PERIOD + 1, the output will never be set/reset depending on POUTC value.

Address: Base address + Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																SAMPLE															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**PWMx\_PWMSAR field descriptions**

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–0 SAMPLE	Sample Value. This is the input to the 4x16 FIFO. The value in this register denotes the value of the sample being currently used.

### 37.7.6 PWM Period Register (PWMx\_PWMPR)

The PWM period register (PWM\_PWMPR) determines the period of the PWM output signal. After the counter value matches PERIOD + 1, the counter is reset to start another period.

$$PWMO\text{ (Hz)} = PCLK(\text{Hz}) / (\text{period} + 2)$$

A value of zero in the PWM\_PWMPR will result in a period of two clock cycles for the output signal. Writing 0xFFFF to this register will achieve the same result as writing 0xFFFE.

A change in the period value due to a write in PWM\_PWMPR results in the counter being reset to zero and the start of a new count period.

**NOTE**

Settings PWM\_PWMPR to 0xFFFF when PWMx\_PWMCR REPEAT bits are set to non-zero values is not allowed.

Address: Base address + 10h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																PERIOD															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0

PWMx\_PWMPR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–0 PERIOD	Period Value. These bits determine the Period of the count cycle. The counter counts up to [Period Value] +1 and is then reset to 0x0000.

### 37.7.7 PWM Counter Register (PWMx\_PWMCNR)

The read-only pulse-width modulator counter register (PWM\_PWMCNR) contains the current count value and can be read at any time without disturbing the counter.

Address: Base address + 14h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

PWMx\_PWMCNR field descriptions

Field	Description
31–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–0 COUNT	Counter Value. These bits are the counter register value and denotes the current count state the counter register is in.



# Chapter 38

## Pixel Pipeline (PXP)

### 38.1 Overview

This document describes the micro-architecture for the proposed Pixel Pipeline used to process graphics buffers or composite video and graphics data before sending to an LCD display or TV encoder.

The goal is to minimize the memory footprint required for the display pipeline and provide an area and performance optimized engine that can meet the needs of both SDRAM-less and SRAM-based systems.

The PXP targets the integration of several independent processing stages into a cohesive strategy to create a pixel pipeline that is flexible enough to handle the requirements of current and future chips.

The PXP combines scaling, color space conversion (or CSC), alpha-blending, secondary color space conversion (or CSC2), pixel conversion lookup memory table (or LUT), and rotation into a single processing engine, as shown in the diagram below. By integrating multiple blocks, intermediate buffer operations to external memory are removed, reducing external memory bandwidth, power, and software control complexity.

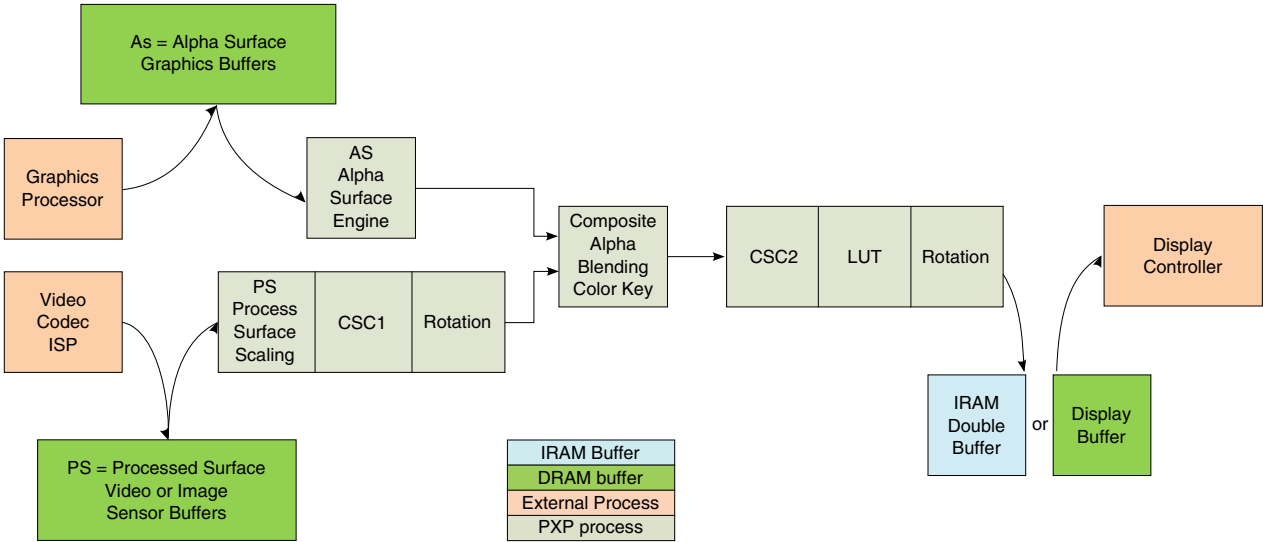


Figure 38-1. PXP Architecture

## 38.2 Clocks

The following table describes the clock sources for PXP. Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

Table 38-1. PXP Clocks

Clock name	Clock Root	Description
clk	pxp_axi_clk_root	PXP clock

## 38.3 Top-level architecture

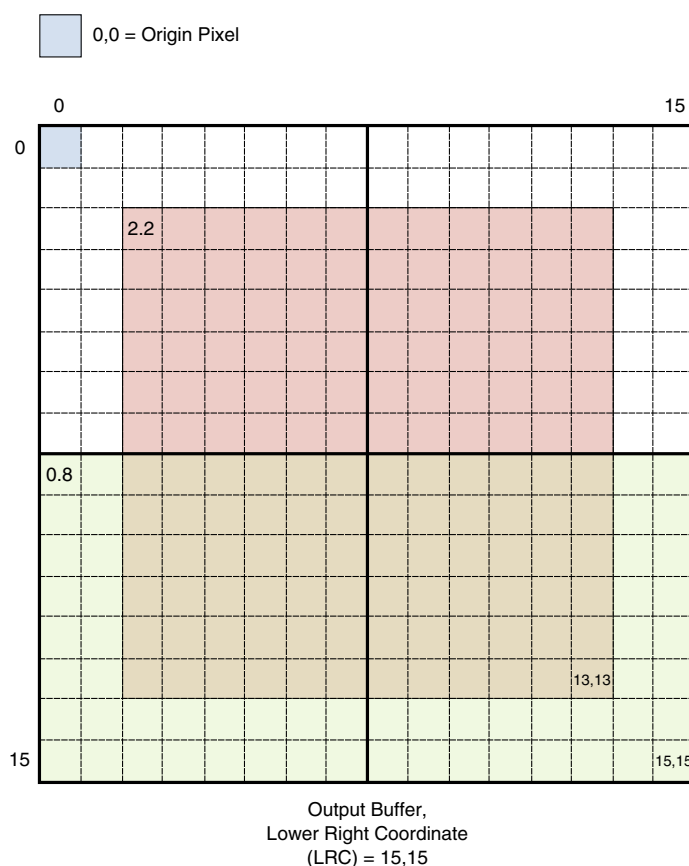
The PXP will consist of several pipelined blocks that perform the video source frame scaling, color space conversion, alpha-blending/color key algorithm, secondary CSC, pixel correction, and input/output rotation.

The entire pipeline will operate within the requirements of the PXP architecture, and will thus perform operations on either 8x8 or 16x16 pixel blocks in the representative source buffers. The entire pipeline will operate within the context of two iteration counters that iterate through the appropriate grid of input blocks to produce the rotated output grid blocks in scan-line order.



Figure 38-1 above shows the high-level architecture of the scaling, color space conversion, blending, pixel correction and rotation engines. The Alpha Formatter fetches one RGB graphics plane alpha surface, or AS. The Scaling engine fetches a single processed surface, or PS, which can be blended with the AS surface. Although the PXP processes NxN pixel macro blocks, each of the AS or PS surfaces can have any pixel alignment within the output buffer. There are no restrictions and any pixel coordinates within the output buffer are valid. The upper left origin of the output buffer is defined as pixel 0,0. The upper left and lower right coordinates for each of the AS and PS surfaces are inclusive within the output buffer.

Figure 38-2 represents a sample output buffer configuration with both an AS and PS surfaces included. The alignment of each AS and PS surface within the output buffer can be at any arbitrary pixel locations. For example, the PS surface has an upper left coordinate (ULC) of 2,2 and a lower right coordinate (LRC) at pixel 13,13. The maximum value for the ULC and LRC for each of the AS and PS surfaces is bounded by the LRC of the output buffer, 15,15 for this example.



**Figure 38-2. Sample output buffer configuration**

The AS engine supports RGB pixel formats, and the PS engine supports RGB, YUV, and YCbCr pixel formats and variants of these pixel types. The CSC1 can be used to convert to RGB pixel formats so that the PS surface can be blended with the AS surfaces in the compositing engine in the RGB color space. There is a single rotation engine in the PXP with a programmable location within the PXP pipeline. Rotation can occur at the output stage after image composition occurs, or it can occur at the output of the PS engine. In the first scenario, all the data produced by the AS and PS engines will be rotated. When the rotation module is programmed to rotate only PS images, the AS is not rotated, and AS pixels are combined with rotated PS surfaces. The CSC2 unit can convert to any color space for final output. Pixels can be corrected using a programmable LUT resource to achieve any desired pixels effects.

### 38.3.1 Processing Details

The PXP architecture has been driven primarily by the requirement that the output buffer must be processed and rotated without intermediate frame buffer stored in external memory.

This reduces the use of external memory bandwidth requirements thus reducing overall system power consumed.

Since the output of the rotation block must be NxN pixel blocks in scan order, the entire pipeline will operate on NxN pixel blocks. In essence, the pipeline will be able to operate on blocks in a random access fashion, but the entire pipeline will operate within the context of two iteration counters that will iterate through the horizontal and vertical input blocks to generate the required output block.

#### Processing Pipeline

The control block will coordinate the processing of the pixel blocks within the source and destination image buffers. It begins by issuing a command to each stage of the pipeline requesting that operations be done for the block at offset x, y. When the block accepts the command, it asserts its acknowledge signal for a single cycle to indicate the acceptance and allow the control unit to move to the next block.

When the PS and AS fetch engines have received a command, they will fetch the required data and place it into their fetch buffers. If compositing the RGB AS surface with the PS surface, then the output of the PS engine needs to be converted to the RGB color space using CSC1, since all compositing occurs in the RGB color space. For YUV output pixel formats, the CSC1 unit can be enabled to convert pixels into the RGB space for subsequent compositing with AS pixels. Then, the CSC2 module can convert the resulting pixels back into the YUV output color space. If the final output color space is YUV and there is no compositing required (AS not present, for example), then both the

CSC units can be bypassed and the pixel data path will pass the YUV pixels to the rotation engine. For YUV output formats, scaling operations, LUT, and rotation operations are still valid, but blending RGB AS surfaces with YUV PS surfaces is NOT supported. The two CSC units in the overall pixel data path must be used to achieve the desired source frame compositing and output pixel formatting.

The alpha blender/color key module will process a pixel any time that both inputs present valid data.

A handshake will be created between each stage and a pipeline controller to handle the advance of the pipeline and generation of the iteration counters. The pipeline controller will also maintain the interlocks with the LCD interface for the case where the LCD display and pixel pipeline use the SRAM to maintain the double buffer block intermediate buffer .

### 38.3.2 Scaling Operation

The scaling engine operates on YUV (or YCbCr) 422 or 420 and any RGB formatted pixels. Each color plane is sourced from color planes indicated by different base address registers.

The scaling source data can be stored as 3 individual planes for each Y, U, and V data, stored as two planes as a single Y and interleaved UV plane, or stored as a single plane with YUV/RGB interleaved on a per byte basis.

The scaled output image is presented to the CSC module as YUV444 or RGB888 pixels with a single byte for each color channel. The scaler can reduce an input image by a maximum factor of 16. In this case, the output image will be 1/16 the dimension of the input image in each of the X and Y axis. There are no limits, essentially, on increasing the source image size. The theoretical maximum increase is 4096 since a 12 bit fractional step function is used when scaling an input image. Scaling in either axis, X or Y is independent, so a source image can appear stretched in either direction.

All source images pass through the scale engine. The PXP alpha blend module and AS pixel streams are in the RGB888 format, so PS pixel buffers must be converted to the RGB888 format for alpha blending. The scaling engine works with the CSC1 module to translate YUV/YCbCr pixel formats to RGB888 for output frame buffer compositing using the alpha blender. The CSC2 module can be bypassed or enabled to convert pixels to any output color space. In the case of processing RGB pixels in the PS engine, the CSC1 unit can be bypassed so compositing can occur in the alpha engine.

The scaling operation is divided into two scaling steps. The first step is a decimation scaler, and the second step is a bilinear filter. The decimation filter provide a maximum down scaling factor of 8, and the subsequent bilinear filter provides a maximum scaling factor of 2. Combined, the maximum scaling factor can be up to 16. The decimation and bilinear scaling engines are independently programmable. There is also an initial offset that is programmable to allow more source data to be considered in the bilinear scaling engine.

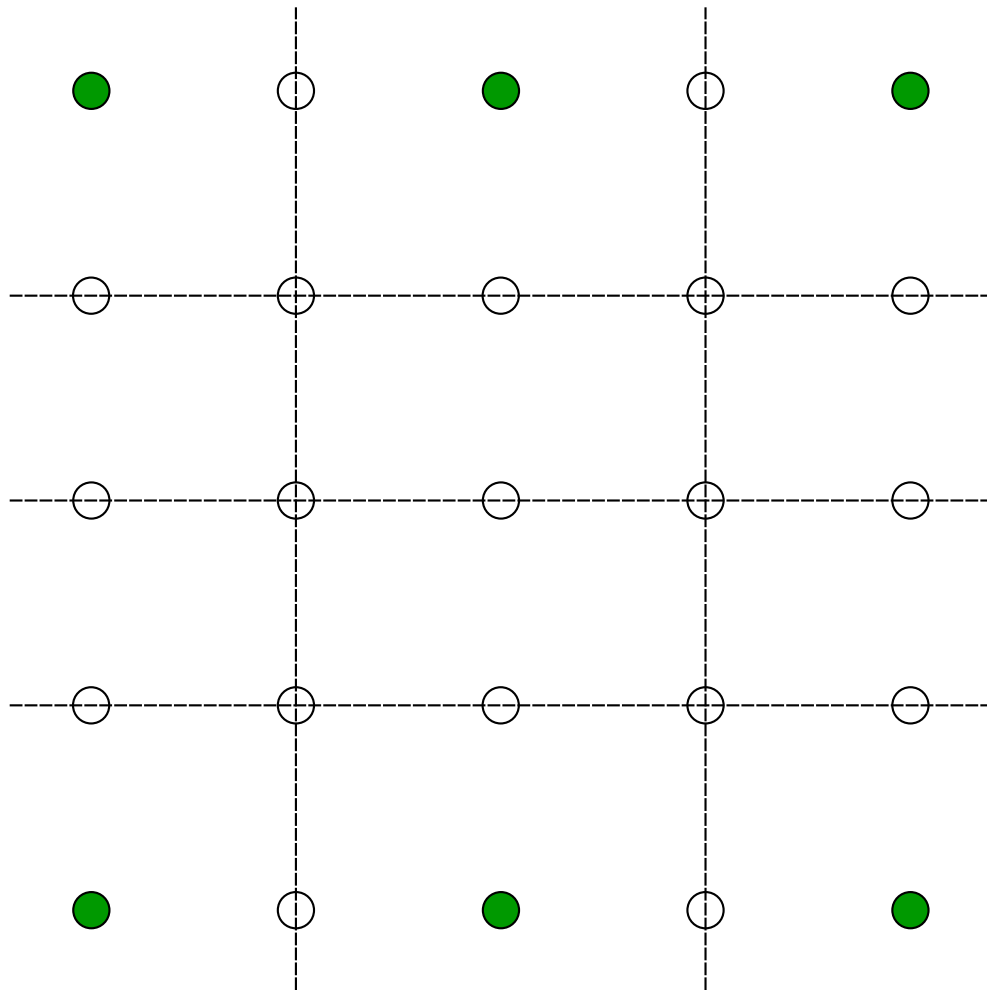
### 38.3.3 Decimation Image Scaling

The first of two scaling engines is the decimation filter.

The intent of the decimation filter is to use as much source data as is possible to create the output image frame buffer. The decimation filter simply discards certain pixels from the source PS image depending on the reduction selected.

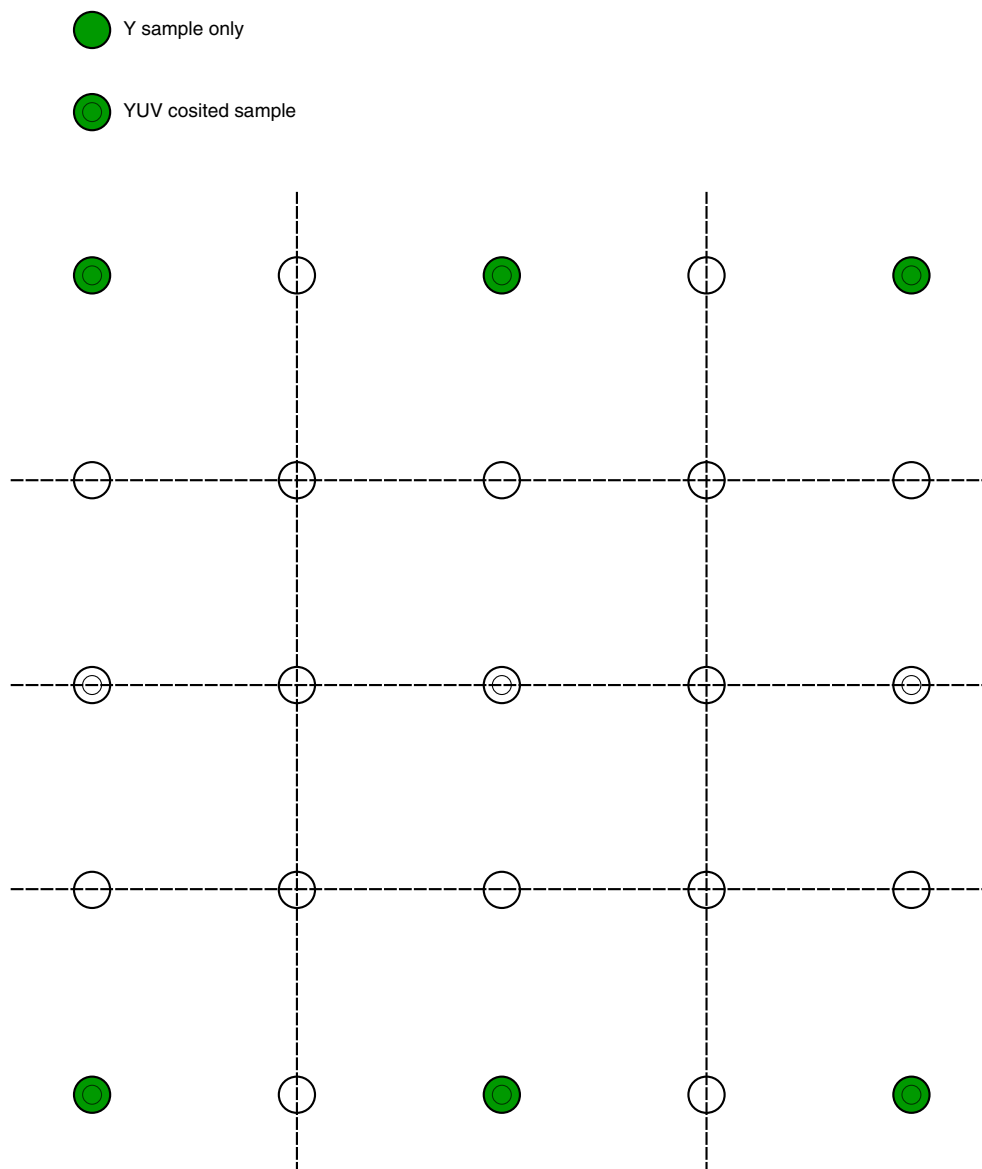
For RGB pixel formats, each color channel is treated equally since there is the same amount of pixel data within each color plane. For YUV422/420 formats, the chroma samples are already subsampled by 2. In these decimation scenarios, the chroma decimation factor is adjusted to account for the pre-decimation of the chroma samples. For example, since YUV422 is already sub-sampled by 2 horizontally, an X decimation factor of 2 does not apply to the YUV422 pixels in the X direction. All the chroma samples are passed on to the bilinear filter in this case. As another example, an X decimation factor of 4 will decimate the chroma samples by 2, since this factor combined with the pre-decimation factor of 2 in the pixel source buffers totals an overall decimation factor of 4.

The following example will show which pixels (in green) in a source RGB buffer that are passed to the bilinear filter for an X decimation factor of 2 and a Y decimation factor of 4. All pixels coincident with dashed lines are discarded.



**Figure 38-3. RGB decimation X /2, Y /4**

Using the same decimation factor as the above scenario for RGB pixels, but using YUV420 source buffers, it can be shown that the decimation factor for the Y and UV components of data are decimated differently. This is due to the pre-decimation of the chroma samples in the source frame buffers. Figure 4: YUV420 decimation X /2, Y /4 indicates that the U/V samples in the X direction are not decimated, but the Y samples in the X direction are decimated by the factor of 2.



**Figure 38-4. YUV420 decimation X /2, Y /4**

### 38.3.4 Bilinear Image Scaling Filter

The PXP implements a bilinear scaling filter to resize an input image to a different resolution for display output.

The bilinear filter is a weighted average of the four nearest pixels that can be sourced to approximate the pixel in the output frame buffer.

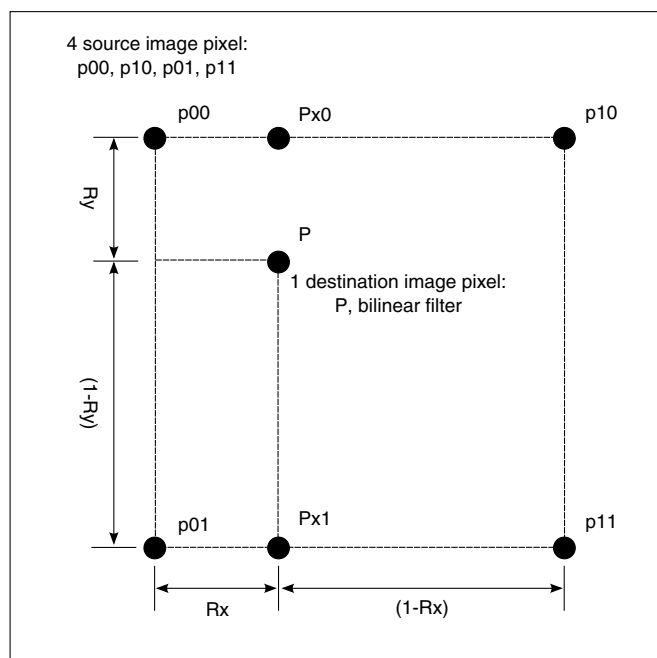
When scaling YUV data, the UV values are offset by 0x80 (top bit inverted) to shift the signed UV bits into an unsigned equivalent with a range of 0 to 255. YCbCr data does not have to be shifted since it is defined as an unsigned byte. The REG\_CSC1\_COEF0[YCBCR\_MODE] bit controls whether this operation is applied to the input UV bytes.

After scaling, the offset is removed so that the range for UV data is signed from -128 to 127.

The reason for this adjustment is based on the implementation of an unsigned scaling engine, and therefore, is to ensure that the scaled values are handled properly. Consider the following table:

Format	pixel0	pixel1	average	Result
decimal	-2	+2	0	Correct
CbCr	0x7E	0x82	0x80	Correct (0x80 is 0 in CbCr)
UV	0xFE	0x02	0x80	Incorrect (0x80 is -128 in UV)
decimal	-32	+16	-8	Correct
CbCr	0x60	0x90	0x78	Correct (0x78 is -8 in CbCr)
UV	0xE0	0x10	0x78	Incorrect (0x78 is +120 in UV)

To compute the output pixel value at position as indicated by P, consider the diagram below.



**Figure 38-5. Output Pixel Value**

A step function is used to indicate the position of the pixel "P" in the output frame. This position may not coincide with a single pixel position in the input frame buffer. In this case, the four closest pixels in the input frame are used to approximate the value of the pixel in the output frame.

The PXP scaler first computes a linear filter in the X axis to create the two intermediate pixel values Px0 and Px1. The step function's X fractional component is used to provide the weighting factor for blending p00 with p10 to provide Px0. Likewise, Px1 is also derived from a linear filter using p01 and p11.

The equations for Px0 and Px1 are as follows:

$$Px0 = p00*(1-Rx) + p10*Rx$$

$$Px1 = p01*(1-Rx) + p11*Rx$$

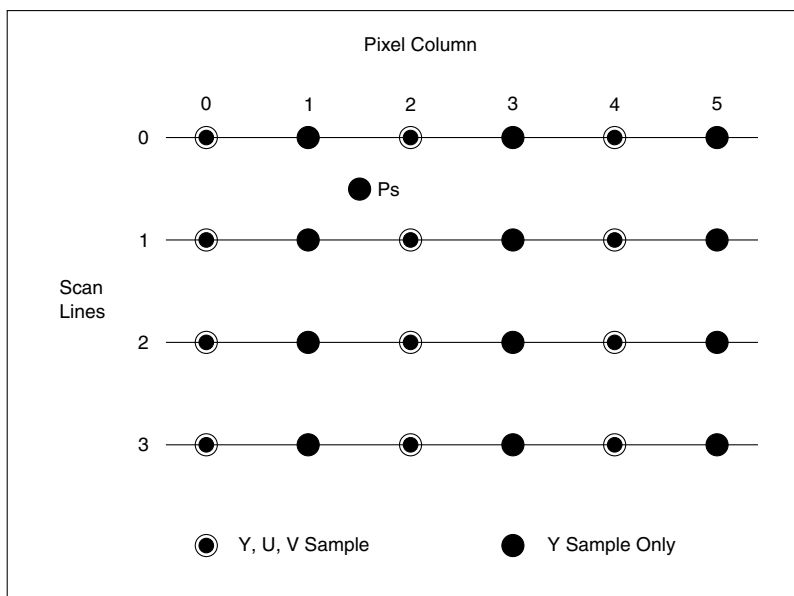
The PXP scaler uses the intermediate X pixels Px0 and Px1 and implements a bilinear filter on these two pixel values to produce the final pixel value at position P. The remainder of the step function for the Y axis is used to compute the weighted average pixel result. The equation for final filtered pixel is:

$$P = Px0*(1-Ry) + Px1*Ry$$

### 38.3.5 YUV 4:2:2 Image Scaling

The following figure illustrates the positioning of YUV samples for the 4:2:2 formats. There are twice as many Y luma samples then U and V chroma samples horizontally.





**Figure 38-6. YUV Sample Positioning, 4:2:2**

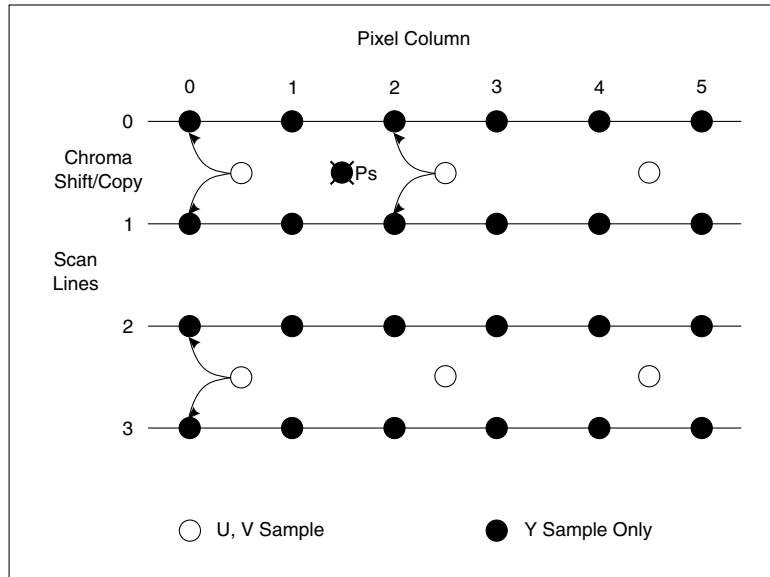
Consider the scaled output pixel  $P_s$  (pixel scaled) which has an accumulated step function of  $X=1.5$  and  $Y=0.5$ . The remainder for the step function is  $R_x = 0.5$  and  $R_y = 0.5$ . Or, the sub pixel position of output pixel  $P_s$  is half way between line 0 and 1 and half way between column 1 and 2.

The Y output component of  $P_s$  is simply the bilinear function of the four nearest Y samples from the input image. Specifically, the Y values at [1,0], [2,0], [1,1], and [2,1] are used to compute the Y for  $P_s$ .

For the U and V components of  $P_s$ , there are no samples present in the column position 1. The bilinear filter uses chroma components located at [0,0], [2,0], [0,1] and [2,1]. Since the chroma components are not sub sampled vertically, the remainder used to combine pixels vertically is  $R_y=0.5$  (the same as for Y). However, horizontally, the scaling engine shifts the remainder by a factor of 2. So an X axis step function value of  $X=1.5$  has a remainder  $R_x=0.75$ . Source chroma values are not replicated, they are completely interpolated using the four nearest chroma samples to approximate U and V at  $P_s$ .

### 38.3.6 YUV 4:2:0 Image Scaling

The following figure illustrates the positioning of YUV samples for the 4:2:0 formats. Chroma is sub sampled both horizontally and vertically. In this format, the chroma frame buffers contain  $\frac{1}{4}$  the data that the luma frame buffers store.



**Figure 38-7. YUV Sample Positioning, 4:2:0**

The Y output component for all scaled pixels in 4:2:0 formats are the same as for the 4:2:2 pixel formats.

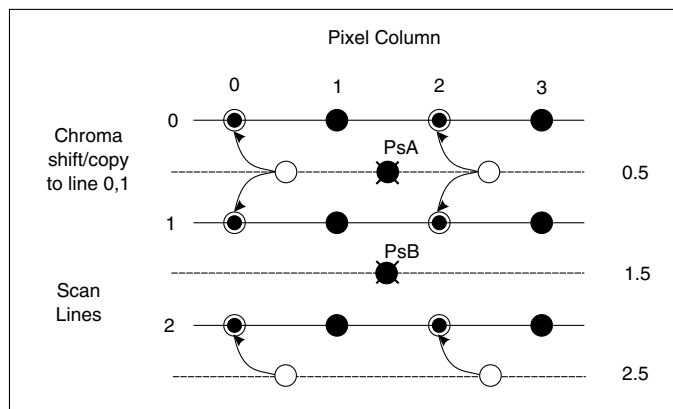
The U and V output components have two considerations when computing the output pixel  $P_s$ .

1. All chroma samples from the input source image are shifted left and up by  $\frac{1}{2}$  a sample position of the input pixel matrix.
2. Odd scan lines are replicated using the previous even chroma scan line values. So, output image chroma values that map between even to odd scan lines are replicated in the vertical axis. In contrast, output image chroma values between odd to even scan lines are interpolated vertically.

The chroma values are interpolated horizontally as in the 4:2:2 pixel format.

As an example, consider the interpolated pixel  $P_s$  in the 4:2:0 diagram above. For the Y component, the interpolated output luma is a function of the Y values in the source frame buffer at position [1,0], [2,0], [1,1], [2,1].

For the U and V interpolated samples, the chroma values on scan line position 0.5 are shifted so that they coincide with the even luma sample points. They are also replicated so that a single chroma scan line is used twice. The chroma scan line at 0.5 is replicated to represent the 4:2:2 sample points for scan line 0 and 1. The chroma scan line at 2.5 is replicated to represent the 4:2:2 sample points for scan line 2 and 3. This pattern of chroma replication occurs for the entire source frame buffer during the scaling operation.



**Figure 38-8. Scaled Chroma Computation Examples**

The preceding diagram has two examples for the computation of the scaled chroma output pixel. For chroma at output position PsA (vertical position 0.5), interpolation occurs in the X axis using chroma values at column 0 and column 2. However, since line 0 and line 1 have equal chroma values due to chroma line replication, scaling in the Y axis results in replication of chroma values.

For chroma at output position PsB (vertical position 1.5), interpolation occurs in both the X and Y axis. The Y axis is an interpolation since the chroma values copied to scan line 1 and 2 and not the same.

In summary, any output image pixels that map to an odd scan line above and an even scan line below are interpolated vertically. Output image pixels that map to an even scan line above and an odd scan line below are replicated vertically.

### 38.3.7 RGB/YUV444 Image Scaling

For all RGB formats, the RGB pixels are converted up to RGB888 with 8 bits per each color component.

Then each color component is passed to the scaling engine and each component is treated in the same manor. The RGB scaling operation is the same as for the Y scaling operation described in the preceding sections. Also, YUV444 contains a byte for each color plane at each pixel location, so all three color components are scaled in the same manor.

### 38.3.8 Color Space Conversion (CSC)

There are two modules in the PXP to convert pixels between color spaces. They are referred to as CSC1 and CSC2 (for lack of a better naming convention).

CSC1 exists after the scaling unit and is dedicated to converting from YUV to RGB. CSC2 is a full duplex color space converter in that it can convert into either RGB or YUV (or YCbCr) color spaces depending on the desired output pixel format. All coefficients are programmed as two's complement numbers and both CSC units can be bypassed if CSC is not desired at either position of these CSC units in the pixel data path.

### 38.3.9 CSC1 Operation

The CSC1 module receives scaled YUV/YCbCr444 pixels from the scale engine and converts the pixels to the RGB888 color space only if CSC1 is enabled.

The CSC1 module will convert only to the RGB color space and it can be bypassed to allow YUV pixels through the data path. These pixels are loaded into the pixel FIFO for processing by subsequent modules in the pixel data path.

The following equations are used to perform YUV/YCbCr → RGB conversion. The constants will be stored in the PXP control registers as two's complement values to allow flexibility in the implementation and to allow for differences in the video encode and decode operations. In addition, this provides a software mechanism to manipulate brightness or contrast.

$$R = C0(Y + Yoffset) + C1(V + UVoffset)$$

$$G = C0(Y + Yoffset) + C3(U + UVoffset) + C2(V + UVoffset)$$

$$B = C0(Y + Yoffset) + C4(U + UVoffset)$$

Note: In the equations above, U and V are synonymous with Cb and Cr in regards to the color space format of the source frame buffer.

Saturation of each color channel is checked and corrected for excursions outside the nominal YUV/YCbCr color spaces. Overflow for the three channels are saturated at 0x255 and underflow is saturated at 0x00.

The table below indicates the expected coefficients for YUV and YCbCr modes of operation:

Coefficient	YUV	YCbCr
Yoffset	0x000	0x1F0 (-16)
UVoffset	0x000	0x180 (-128)
C0	0x100 (1.00)	0x12A (1.164)
C1	0x123 (1.140)	0x198 (1.596)
C2	0x76B (-0.581)	0x730 (-0.813)
C3	0x79B (-0.394)	0x79C (-0.392)
C4	0x208 (2.032)	0x204 (2.017)

### 38.3.10 YUV versus YCbCr Support

By default, the PXP color space coefficients are set to support the conversion of YUV data to RGB data.

If YCbCr input is present, software must change the coefficient registers appropriately (see the register definitions for values). Software must also set the YCBCR\_MODE bit in the COEFF0 register to ensure proper conversion of YUV versus YCBCR data.

### 38.3.11 CSC2 operation

The CSC2 module receives pixels in any color space and can convert the pixels into any of RGB, YUV, or YCbCr color spaces.

All coefficients are programmable and in the two's complement notation. The output pixels are passed onto the LUT and rotation engine for further processing.

The following equations indicate the CSC2 modules ALU architecture.

Selecting RGB output in REG\_CSC2\_CTRL[CSC\_MODE] configures the ALU in the following manor:

$$R = A1(Y-D1) + A2(U-D2) + A3(V-D3)$$

$$G = B1(Y-D1) + B2(U-D2) + B3(V-D3)$$

$$B = C1(Y-D1) + C2(U-D2) + C3(V-D3)$$

Selecting YUV output configures the ALU in the alternate manor:

$$Y = A1*R + A2*G + A3*B + D1$$

$$U = B1*R + B2*G + B3*B + D2$$

$$V = C1*R + C2*G + C3*B + D3$$

Saturation of each color channel is checked and corrected for excursions outside the nominal color space. Overflow for the three channels are saturated at 0x255 and underflow is saturated at 0x00.

### 38.3.12 Alpha Blending/Color Key

Regardless of pixel input format, the PS and AS pixels are normalized to 32-bits, organized as one alpha and three data bytes.

Alpha blending occurs in the RGB space, if blending is required, PS pixels should be converted to RGB space. If no alpha blending is required, then YUV pixels can bypass the alpha blending ALU without color space conversion.

All pixels are processed by the pixel ALU, but the ALU operations can be disabled to achieve pixel pass through for either PS or AS source pixels.

### 38.3.13 Alpha Blend

The alpha value for an individual pixel represents a mathematical weighting factor applied to the AS pixel.

An alpha value of 0x00 corresponds to a transparent pixel and a value of 0xFF corresponds to an opaque pixel.

The effective alpha value for an AS pixel is determined by the REG\_AS\_CTRL[ALPHA] and REG\_AS\_CTRL[ALPHA\_CTRL] register fields. If

REG\_AS\_CTRL[ALPHA\_CTRL] = ALPHA\_OVERRIDE, the alpha value for the pixel is taken from the REG\_AS\_CTRL[ALPHA]. This can be useful for applying a constant alpha to an entire image or for image formats that don't include an alpha value. If REG\_AS\_CTRL[ALPHA\_CTRL] = ALPHA\_MULTIPLY, the pixel's alpha value will be multiplied by the pixel's ALPHA value in order to allow scaling of the pixel's alpha or to provide better control for pixel formats such as RGB1555, which only contains a single bit of alpha.

For each color channel, the equation used to blend two source pixels is defined below:

$Gá$  = PIO programmed global alpha (8-bit value).

$Eá$  = Embedded alpha associated with AS pixel.

$$á = Gá * Eá + 0x80$$

The result for the red channel as an example:

$$R[7:0] = (á * PS.r) + ((1 - á) * AS.r)$$

When  $á$  is 0xff, the PS pixel will not be blended with the AS pixel, but PS will be passed as the output pixel and will not be blended with AS. In this case, AS will be discarded. Likewise, if  $á$  is 0x00 for a given pixel, PS will be loaded as the output pixel.

REG\_AS\_CTRL[ALPHA\_INVERT] provides the option to invert the final alpha value. This essentially inverts the effect the alpha value has on the AS and PS blending operation.

### 38.3.14 Color Key

The color key function is provided to create transparent effects on the output pixel.

Color keying is applied on the input pixels after they are converted to 8-bits for each red, green, and blue color channels (color keys are not applied directly to 16-bit pixel formats but to their corresponding 24-bit representation). A color key range is programmable for both PS and AS pixels. If the PS 24-bit pixel is within the PS color key range, then AS is passed through the pixel pipeline. In this case, alpha blending does NOT occur.

Conversely, if PS is within the AS color key range, then PS is passed via the PXP data pipeline. If both PS and AS color key tests pass, then the back ground color register is passed onto following PXP processing components in the pipeline.

The condition for color keying to be satisfied is:

$$CK0.r.low \leq PS.r \leq CK0.r.high$$

$$CK0.g.low \leq PS.g \leq CK0.g.high$$

$$CK0.b.low \leq PS.b \leq CK0.b.high$$

For example, if the "red" 8-bit value for the PS pixel (or PS.r) is between the color key low and high values (CK0.r.l and CK0.r.h), the condition is true for the red color plane. When ALL three color planes meet this condition, then only the PS pixel is loaded into the output register.

To disable color keying, program the low color key register value to 0xff and the high value to 0x00. This will guarantee that the color key range test will never be true.

### 38.3.15 LUT

The lookup table (LUT) is used to modify pixels in a manner that is not linear and that cannot be achieved by the color space conversion modules.

Nonlinear response to the input pixels can be achieved based on how the lookup table is programmed.

Programming of the direct access LUT table can be facilitated by single PIO register writes or DMA access. For efficient loading of the LUT, DMA access should be used.

### 38.3.16 Lookup Modes

The LUT has four lookup modes. The lookup modes determine how the `src_pixel` is used to address the LUT memory.

The four lookup modes are:

1. `DIRECT_Y8`
2. `DIRECT_RGB444`
3. `DIRECT_RGB454`
4. `CACHE_RGB565`

The `DIRECT` modes access the LUT memory as a monolithic SRAM. The `CACHE_RGB565` will access the memory as a 2-way set associative cache.

### 38.3.17 `DIRECT_Y8`

`DIRECT_Y8` is used for a 256-byte lookup. In `DIRECT_Y8`, the most significant byte of the pixel is used to address the LUT entry.

This byte reflects the Y/R channel of the pixel data path. Luma, or monochrome, transformations are possible with this lookup mode. The address is generated as: `src_pixel[23:16]`. In `DIRECT_Y8` operation, the memory is byte addressable.

### 38.3.18 `DIRECT_RGB444`

`DIRECT_RGB444` is used for a 8KB (4K pixel) RGB444 to RGB565 lookup.

Pixel formats that are in the YUV color space at the position of the LUT in the PXP data path can also be converted. To take advantage of the full 16KB memory, the `REG_LUT_CTRL[SEL_8KB]` bit can be used to select the upper or lower 8KB memory, thus facilitating the use of 2 separate 444 LUT tables. In `DIRECT_RGB444`, the `src_pixel` is RGB/YUV[23:0] data is used to generate the lookup address. The address is generated as: `{R/Y[23:20],G/U[15:12],B/V[7:4]}`. In `DIRECT_RGB444`, the memory is pixel (2 byte) addressable.

### 38.3.19 `DIRECT_RGB454`

`DIRECT_RGB454` is used for a 16KB (8K pixel) RGB454 to RGB565 lookup.



Pixel formats that are in the YUV color space at the position of the LUT in the PXP data path can also be converted. In DIRECT\_RGB454, the src\_pixel is RGB/YUV[23:0] data is used to generate the lookup address. The address is generated as: {R/Y[23:20],G/U[15:11],B/V[7:4]}. In DIRECT\_RGB454, the memory is pixel (2 byte) addressable.

### 38.3.20 CACHE\_RGB565

The CACHE\_RGB565 lookup is used for a 128KB (65K pixel) RGB/YUV565 to RGB/YUV565 lookup.

The 128KB memory requirement is too costly from an area perspective to implement a complete lookup table in the LUT's on chip memory. For this reason, a LRU (least recently used) 16KB 2-way set associative cache has been implemented to reference the full 128KB lookup table stored in external memory.

The 2-way set associative cache is organized in the following way:

- 16KB total data storage
- 512 entries split between 256 ways
- 6 pixels/entry cache line

Cache efficiency is very critical. For a cache miss, the PXP will be stalled until the cache line can be filled. For a 32 bit DDR memory interface, the latency can be calculated as follows:

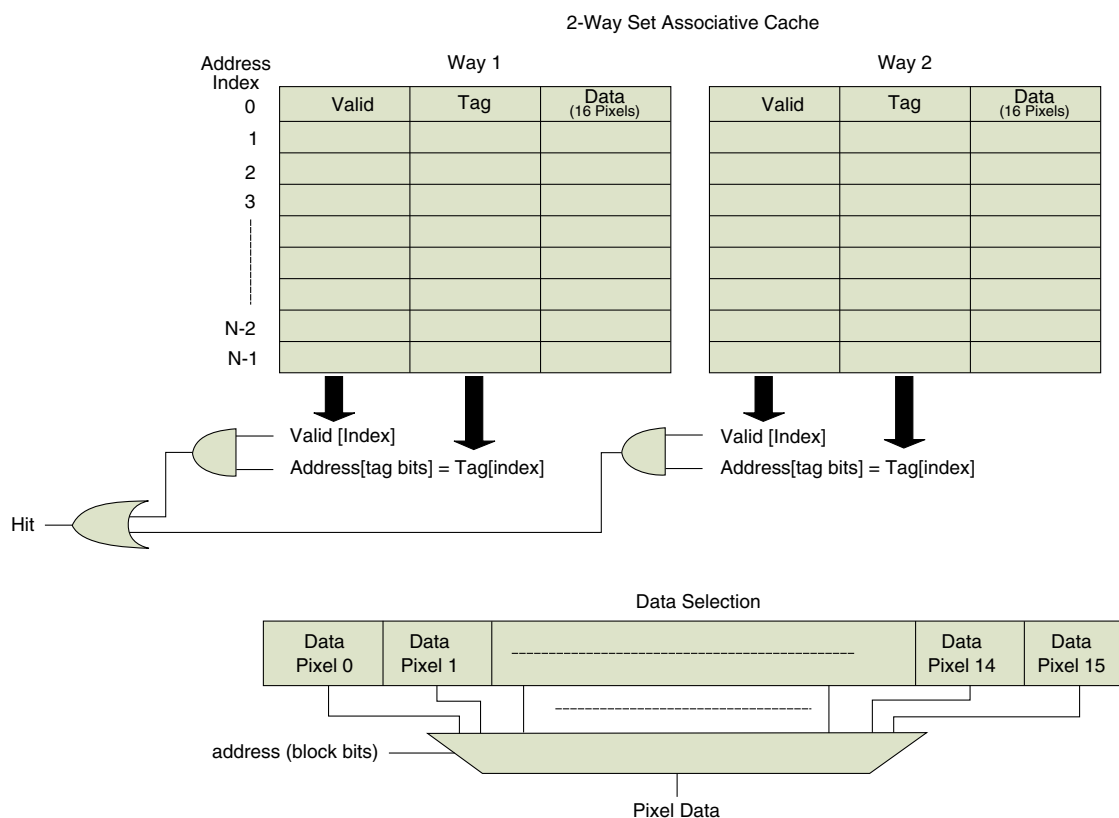
# cycles latency for read command through DDR controller + CAS Latency + # burst cycles for read data to be returned + # cycles latency for data through DDR controller to LUT + 1 cycle for cache access of new data

The src\_pixel is RGB[23:0] data used to generate the lookup address. To improve the cache efficiency the RGB/YUV565 address and the lookup table must be formatted in the following way:

Address:  $A[15:0] = R_7G_7G_6B_7 R_6G_5B_6 R_5G_4B_5 R_4G_3B_4 R_3G_2B_3$

The address is organized as follows:

- A[15:12] tag address, used to compare a hit between cache ways
- A[11:4] index address, used to select cache set (i.e. row of memory)
- A[3:0] block address, used to select Pixel in the cache line



**Figure 38-9. 2-Way Set Associative Cache and Data Selection**

### 38.3.21 Output Modes

The LUT has three output modes for color space conversion.

1. Y8
2. RGBW4444CFA
3. RGB888

### 38.3.22 Y8

With `out_mode` set to Y8, in conjunction with `DIRECT_Y8` lookup mode, the intended operation is Gamma Correction.

Only the third byte is processed by the lookup table. The third byte is represented by the Y value or the R value in the data path since pixel data is either YUV[23:0] or RGB[23:0] where the Y or R byte encompass bits [23:16] respectfully. So, bits 15:0 are

always bypassed and left unchanged. Currently, the LUT is intended to process Y data when any YUV, Y8, or Y4 output pixel formats are selected. However, this resource can be enabled and used for any conceivable purpose.

Note: When the DIRECT\_RGB444, DIRECT\_RGB454 or CACHE\_RGB565 lookup mode is selected in conjunction with the Y8 output mode the low order byte of the two bytes read from the LUT memory will be used as the Gama Correction value.

### 38.3.23 RGBW4444CFA

With REG\_LUT\_CTRL[OUT\_MODE] set to RGBW4444CFA, the REG\_CFA[DATA] is used to select one nibble from the LUT 16 bit output value.

The LUT memory lookup will contain a RGBW4444 value. The REG\_CFA[DATA] will select the R,G,B or W nibble as the pixel value to present to the PXP data path based on the matrix defined by the REG\_CFA[DATA] register. The 4 bit value is presented in the Y, or third byte lane, of the PXP data path. The final pixel transferred to the next PXP stage is {CFA[3:0],CFA[3:0],LUT[15:0]}

#### 38.3.23.1 CFA Correction

The 32-bit REG\_\_CFA[DATA] register is used to encode 16 CFA correction values.

The CFA correction values are encoded as follows:

- 00 selects R
- 01 selects G
- 10 selects B
- 11 selects W

The CFA correction uses 4x4 block processing. Figure 5; CFA mapping translation shows how CFA 4x4 blocks will iterate over the PXP's 8x8 or 16x16 pixel block being processed:

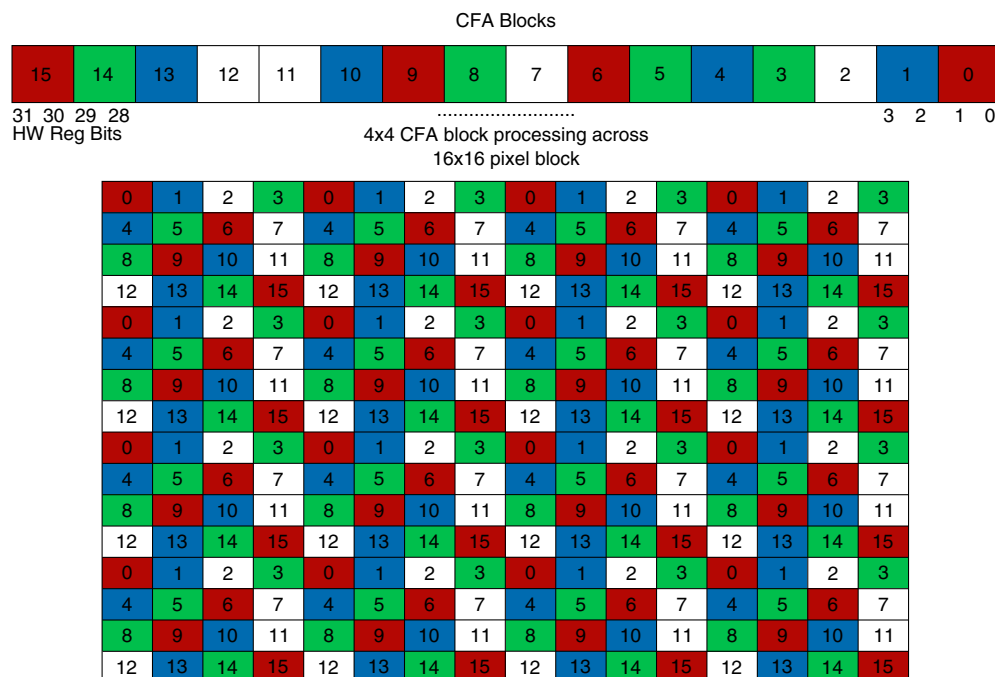


Figure 38-10. CFA mapping translation

### 38.3.24 RGB888

With `out_mode` set to RGB888, the memory output data is interpolated from RGB565 to RGB888.

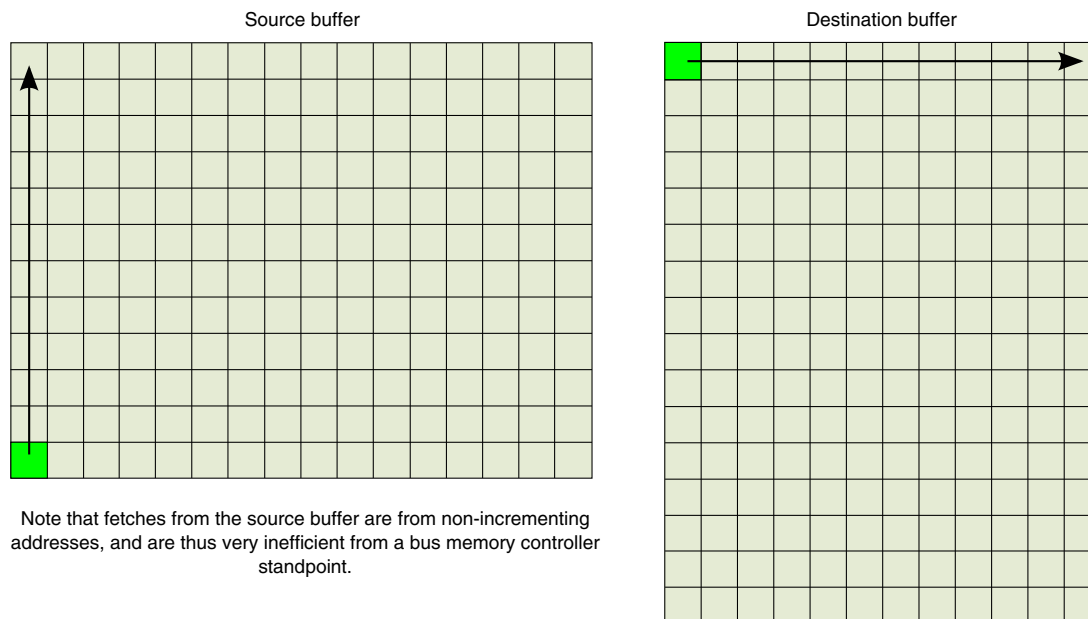
The RGB[23:0] data is formatted as follows: R[7:3]R[7:5],G[7:2]G[7:6],B[7:3]B[7:5].

### 38.3.25 Rotation

There is a single rotation resource integrated into the PXP. The location of this resource within the PXP data path is programmable. >Rotation can occur after compositing the AS and PS buffers in the output stage.

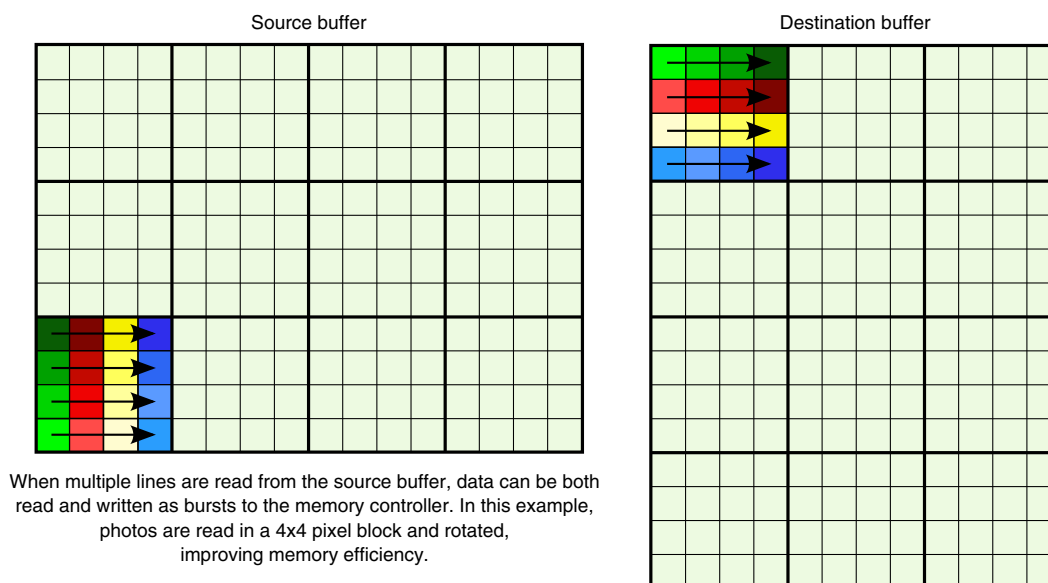
As an alternative configuration, the PS buffer can be rotated and later composited with the AS surface that is not rotated. There is a single configuration bit that provides the configuration of where rotation is implemented within the PXP.

To rotate graphics, the hardware must read pixels in one direction across a frame buffer and write them in an alternate orientation. For the 90 and 270 degree cases, this means that lines of pixels must either be read or written vertically in a frame buffer.



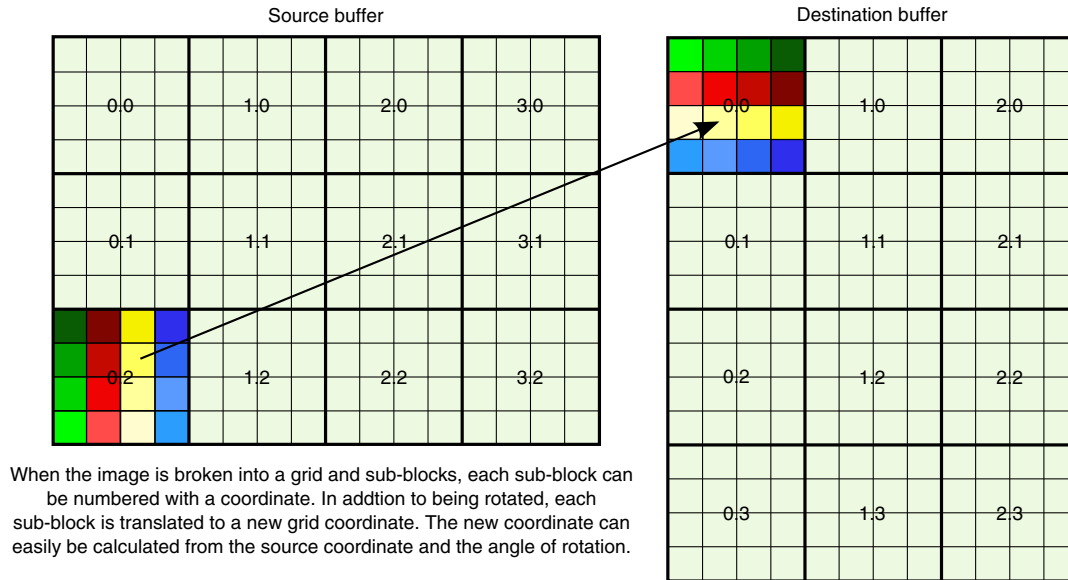
**Figure 38-11. Rotation Read and Write**

In order to rotate efficiently, multiple columns must be rotated to enable the engine to both fetch and store bursts of pixels, thus improving memory performance. The simplest method of doing this is to operate on square blocks of pixels. To rotate the image, each sub-block of pixels must be rotated by the required rotation angle.



**Figure 38-12. Rotated Sub-blocks**

To manage the rotation process, the source image can be broken into a grid of sub-blocks that have coordinates as shown in the diagram below. In addition to rotating the sub-block, each block must be translated to a new coordinate location. For each of the rotation angles (0, 90, 180, 270), it is possible to define a simple algorithm for computing the new translated grid address. The hardware must then simply compute the memory address from the base grid address for both load and store operations.



**Figure 38-13. Grid of Sub-Blocks with Coordinates**

In order to balance the requirements of reasonable burst sizes to the memory controller as well as keep the hardware storage requirements to a minimum, the blending/rotation engine will operate on either 8x8 or 16x16 pixel blocks.

**IMPORTANT NOTE:** An important artifact of the PXP is when rotating a source image and the output is NOT divisible by the block size selected. The output engine essentially truncates any output pixels after the desired number of pixels has been written. Since the output buffer is written as a horizontal row of blocks, the incorrect pixels could be truncated and the final output image can look shifted. In the case where the block size is programmed to 8x8, and the output size that is programmed is 12x12, then there is a remainder of 4 pixels that will be truncated in either the X and/or Y axis when the PXP operation is complete. The output will be shifted by 4 pixels in this example. To compensate for this, the source base address needs to be adjusted so the correct pixels get truncated and the image does not look shifted. In this example, with 90 degrees of rotation, the PS base address should be adjusted by 4 times the actual PS base address - (4\*pitch).

### 38.3.26 Output Buffer

The output buffer engine accepts data from the PXP pixel pipeline and issues requests to transfer the output pixels to external DRAM or the internal SRAM double buffer row of blocks.

### 38.3.27 Address calculator

Each of the blocks will manage its own fetch address using a common address calculator block that computes real addresses from a base address and relative block offset from the base.

Each block will then perform the multiple line fetches (or stores) required to perform the operation. This hides all the address buffer computations from the processing blocks and allows each block to simply track the coordinate of the block it is working on.

### 38.3.28 Block size selection

The PXP can be configured to process blocks that are either 8x8 pixels or 16x16 pixels with the REG\_CTRL[BLOCK\_SIZE] control bit.

When selecting a 16x16 pixel block size, the accesses to fetch AS and PS images and write the final frame buffer are more efficient since twice as much data is requested and processed per memory request.

When optimizing the system for memory bandwidth and image processing time, configure the PXP to process 16x16 pixel blocks.

### 38.3.29 Interlaced Video Support

The PXP has some minimal ability to generate interlaced video content from a progressive source. There two available options, based on the bandwidth requirements and how software is managing video frames.

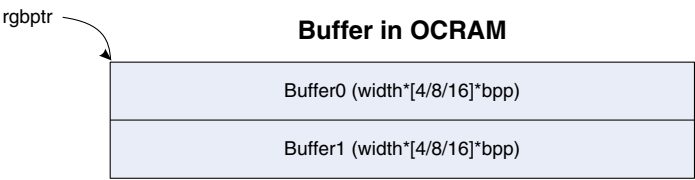
The PXP can either interlace on the input side (by reading every other line of input data) or on the output side (by writing the individual lines of video into two separate fields). Generally, output interleaving should be used since it is the most flexible mode (it allows scaling and full overlay support) and it only requires a single pass of the PXP to generate two separate output fields.

Input interleaving can be beneficial in cases where the PXP is running at 60fps, since it requires fewer fetches to produce the output data. There is no direct hardware support for input interleaving, in that, there is no configuration bit that can be set to alter how the PXP processes a frame for input interleaving. Input interleaving is achieved by simply setting the source frame buffer pitch value to twice the value it would normally be set to for the equivalent progressive frame. The output parameters also need to be consistent with the desired processing effect. For example, the vertical resolution would be set to account for the reduced resolution to process the interlaced input buffers.

### 38.3.30 LCDIF Handshake

The PXP and LCDIF support a mode where the internal SRAM can be used for the frame buffer to minimize external memory bandwidth required.

This is accomplished by creating two buffers in SRAM, a double buffer row of blocks, where each correspond to 8/16-lines of the frame buffer. The buffers must be consecutive and allocated as a single block of data.



**Figure 38-14. Buffer in OCRAM**

The storage required can be calculated for an 8x8 block size as

- $\text{storage} = 16 (\text{lines}) * \text{rotated\_row\_length} * \text{pixel\_size}$

and for 16x16 block size as

- $\text{storage} = 32 (\text{lines}) * \text{rotated\_row\_length} * \text{pixel\_size}$

where  $\text{pixel\_size} = 4$  for 32bpp or 2 for 16bpp modes. The following table lists the storage requirements for common image sizes using 8x8 block size:

Image Size	Storage (16bpp)	Storage (24bpp)	Storage (32bpp)
320x240 (QVGA) - 0/180 rotation	10KB	15KB	20KB
320x240 (QVGA) - 90/270 rotation	7.5KB	11.5KB	15KB
640x480 (VGA) - 0/180 rotation	20KB	30KB	40KB

*Table continues on the next page...*



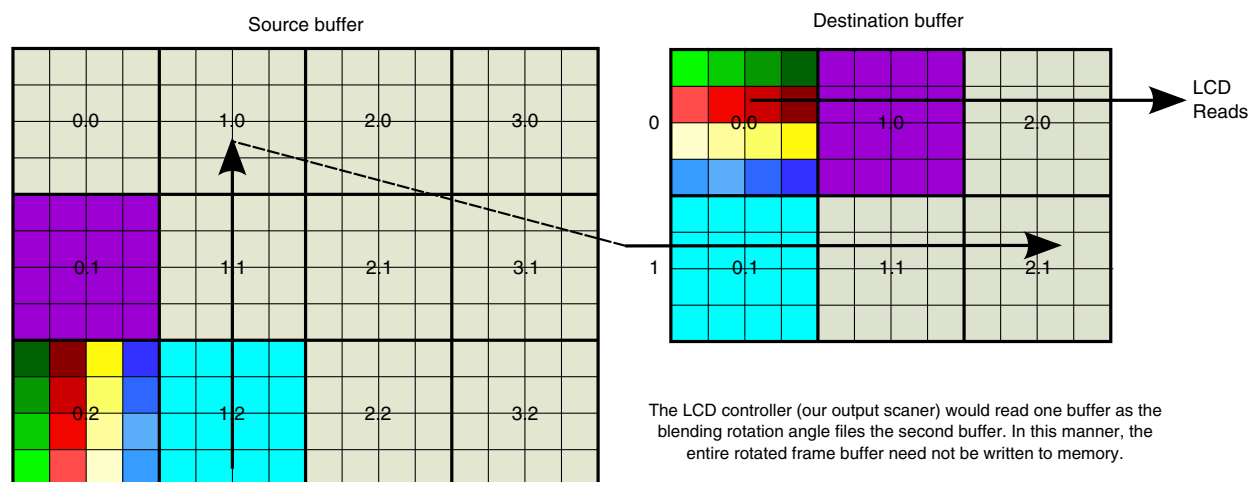
640x480 (VGA) - 90/270 rotation

15KB

22.5KB

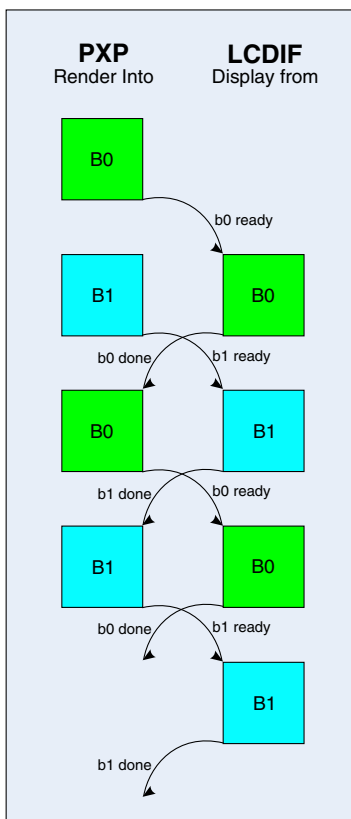
30KB

The following diagram shows how the minimal rotation buffer would be organized. As the engine and LCD progress down the image, they continually swap roles of filling and emptying each eight-line buffer.



**Figure 38-15. Minimal Rotation Buffer Organization**

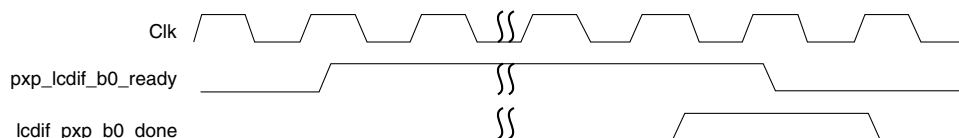
When this mode is enabled, the PXP will process one row of pixel blocks and write the results to the first SRAM buffer (buffer 0). The PXP will then alternate between writing subsequent rows to buffer 0 and buffer 1. After the PXP generates the data for one buffer, the LCDIF will begin reading that buffer and send the contents to the display device. Once the LCDIF finishes reading a buffer, it will start displaying from the other buffer while the PXP continues filling the previously processed buffer.



**Figure 38-16. PXP and LCDIF Buffer Sharing**

To accomplish the buffer sharing, the PXP and LCDIF will maintain buffer status using a pair of handshake signals. When a buffer is filled by the PXP, it will assert the `pxp_lcdif_bx_ready` (where x is 0 or 1) signal to indicate to the LCDIF that the buffer has valid data. The LCDIF will then release the buffer by asserting the `lcdif_pxp_bx_done` signal.

The basic protocol is shown in the diagram below:



**Figure 38-17. Buffer Sharing Protocol**

The PXP will continue to assert the `bx_ready` signal until the corresponding `bx_done` signal is sampled high for one clock cycle. It will then deassert the `bn_ready` until the next time the buffer has been filled. After the PXP samples the `bx_done` signal asserted, it is free to begin filling the buffer with the next block size lines of display data. If a buffer has not been released when the PXP is ready to process data for that buffer, it will suspend rendering operations until the buffer has been released by the LCDIF.

### 38.3.31 LCDIF Abort

When the memory subsystem is not loaded, the PXP should be able to render the buffers faster than the LCDIF can drain the buffers.

It is possible under some scenarios (high LCDIF output rates with high memory latency) that the PXP may not be able to keep up with the LCDIF, even in the SRAM mode of operation. When this happens, the LCDIF will signal that it has completed one of the buffers before the other has been rendered by the PXP. This condition will be detected by the PXP's control logic as an "LCDIF Abort", which will cause the PXP to abort processing in the current row and proceed to the following row. It will acknowledge the abort to the LCDIF by raising the `buffer_ready` signal for the current buffer to enable the LCDIF to begin displaying the partially-filled buffer. While an abort will create artifacts in the video display, it does minimize the artifacts by limiting them to the remaining pixels blocks in the current row versus ruining the entire frame buffer.

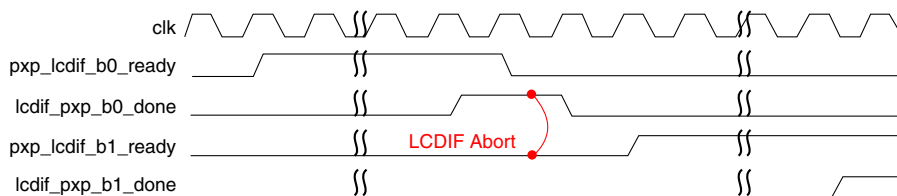


Figure 38-18. LCDIF Abort

### 38.3.32 Theory of Operation

The PXP can be used to accelerate graphics operations by offloading graphics processing from the processor. The block can perform alpha blending and color key substitution on two RGB graphics buffers.

The PXP is organized as having a processed surface (PS) and an alpha surface (AS) that can be blended with the processed surface. There are no restrictions on the location of the AS or PS within the output surface (OS). As the PXP processes NxN blocks, operations

are performed on a pixel by pixel basis. The AS and PS pixels are alpha blended, color keyed, process by CSC and LUT resources as individual pixel components. This allows efficient block processing with supporting arbitrary alignment for both the AS and PS surfaces. The resulting pixel block is then written to the corresponding block in the output buffer.

### 38.3.33 Pixel Handling

All pixels are internally represented as 24-bit values regardless of input or output pixel formats.

The pixels get converted in the AS and PS buffer engines to 24-bit pixel values.. There is also an 8-bit alpha value at stages up to the alpha blender within the PXP for blending within the RGB color space. Compositing of AS and PS images can only occur in the RGB color space. If compositing is not required, then YUV pixels can be transferred and processed at all PXP pixel resource components.. The color orientation of pixels within the PXP can be controlled by the CSC1, CSC2, and LUT resources.

For RGB, input pixels are converted into 24-bit pixel values using the following rules:

1. 32-bit ARGB8888 pixels are read directly with no conversion for both the AS and PS.
2. 32-bit RGB888 pixels are assumed to have an alpha value of 0xFF (full opaque).
3. 6-bit RGB565 and RGB555 values are expanded into the corresponding 24-bit color space and assigned an alpha value of 0xFF (opaque). The expansion process replicates the upper pixel bits into the lower pixel bits (for instance a 16-bit RGB555 triplet of 0x1F/0x10/0x07 would be expanded to 0xFF/0x84/0x39).
4. 16-bit RGB1555 values are expanded into the corresponding 24-bit color space and assigned an alpha value of either 0x00 or 0xFF, based on the 1-bit alpha value in the pixel. The ALPHA\_MULTIPLY function is useful in this scenario to allow scaling of the opaque pixels to a semi-transparent value.

Alpha values can be passed through the entire PXP data path and output in ARGB888 and ARGB555 pixel modes. Also, output pixels can be assigned an alpha value using the REG\_OUT\_CTRL[ALPHA] register. 16-bit pixels values are formed from the most significant bits of the 24-bit pixel values.

When YUV/YCbCr output formats are selected, all pixels are internally represented as either RGB or YUV pixels values. The CSC2 or LUT can convert internal RGB/YUV pixels into the correct output format. In this way, any PS color space can be blended with AS RGB pixels and output in any color space.

### 38.3.34 Output Buffer Composition

The output buffer will be rendered by composing each pixel block from the associated PS and AS buffers.

The AS pixel buffer can be blended or color-keyed with the associated data from the PS buffer (either the PS image pixels or REG\_PS\_BACKGROUND register based on PS programmed coordinates).

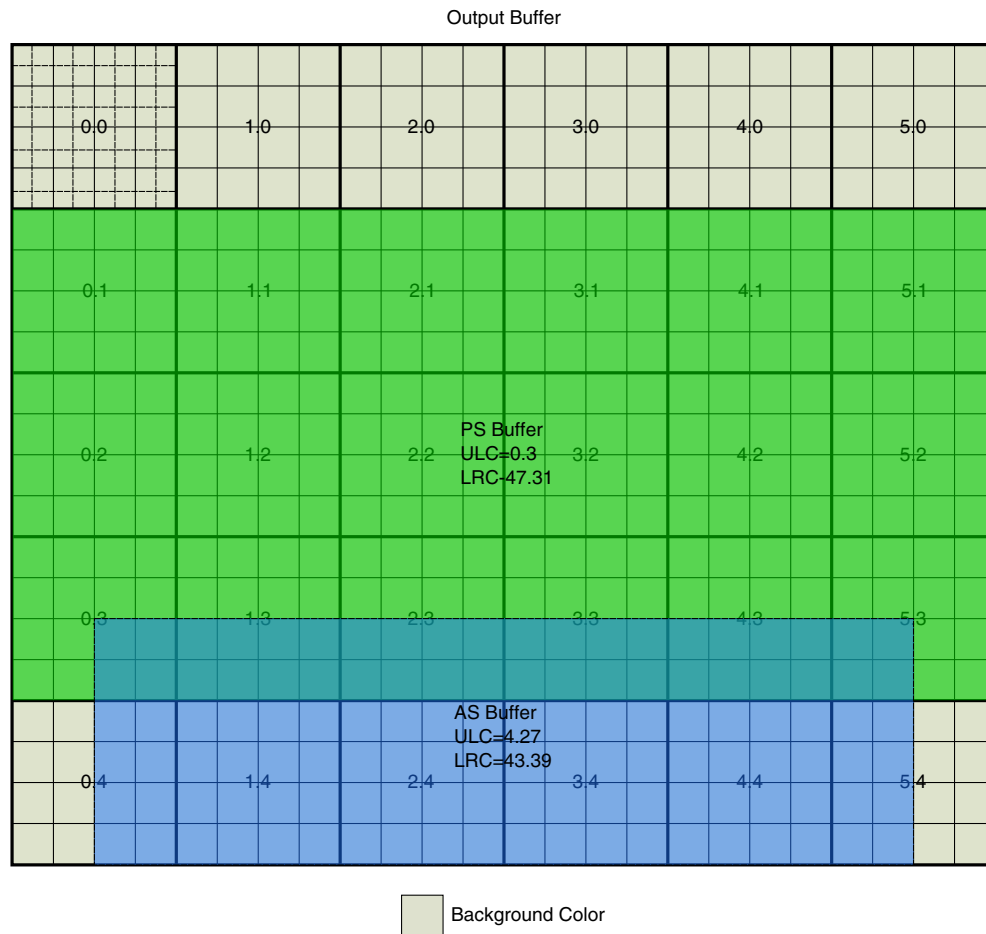


Figure 38-19. Output Buffer Composition

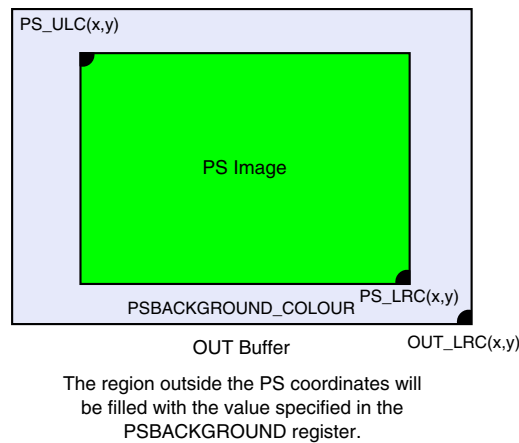
### 38.3.35 PS Image Processing

As the PXP processes image buffers, it iterates over the output buffer by fetching the corresponding input buffer blocks and processing the pixels embedded in these.

### 38.3.36 Letterboxing

At each pixel coordinate, the control logic determines if the PS pixel (argument also applies to AS pixels) will be used in rendering the output pixel.

This is determined by checking the output pixel's coordinates against the REG\_OUT\_PS\_ULC and REG\_OUT\_PS\_LRC (ULC and LRC in short) register contents. For pixels outside this region, the PS pixel will be loaded with the pixel value from REG\_PS\_BACKGROUND, which can be used to effectively control the letterboxing color. There are no block size or block boundary restrictions when setting the ULC or LRC for either the AS or PS. The only restriction is that the ULC and LRC are within the OUT LRC extents.



**Figure 38-20. OUT Buffer**

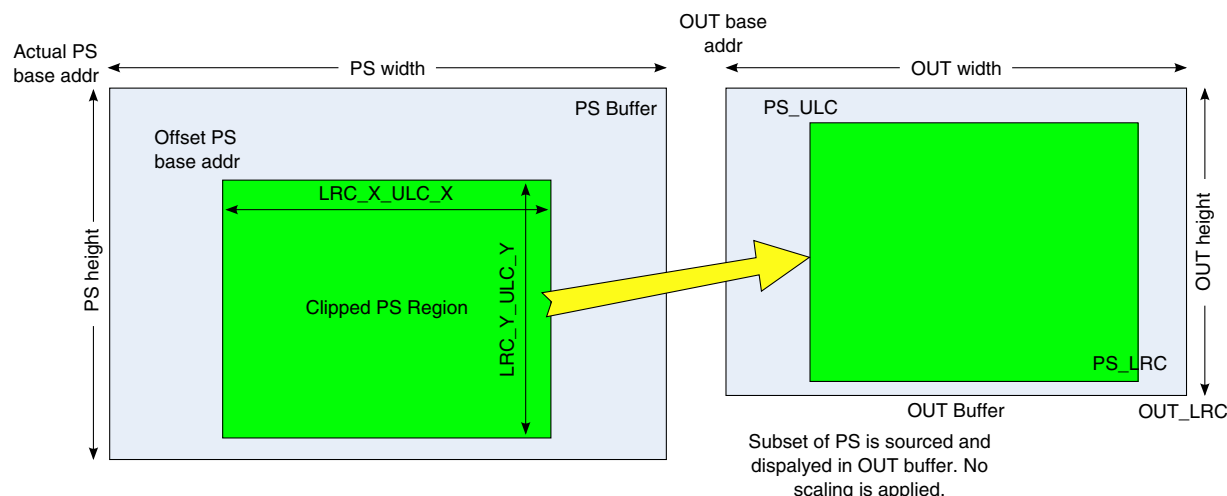
### 38.3.37 Clipping source images

A subset of the PS buffer can be used in rendering the output buffer. The PXP\_PS\_BUF register can indicate an offset into the PS buffer that will be used for display within the OUTPUT buffer.

The pixel at the address defined in the PXP\_PS\_BUF register will be the pixel that is displayed at the pixel coordinate indicated by PXP\_OUT\_PS\_ULC within the output buffer. Essentially, the PXP\_PS\_BUF register can be used to establish an offset into the PS buffer thus clipping all PS buffer pixels that are at a lower address. The PXP\_PS\_PITCH will always indicate the number of bytes that are vertically adjacent in

the PS buffer. The settings in the PXP\_PS\_BUF, PXP\_OUT\_PS\_ULC, and PXP\_OUT\_PS\_LRC will determine the subset of the PS buffer, or clipped PS source buffer, that will be used in the output buffer.

It is important to note that when scaling the PS buffer, the coordinates of the PS buffer within the output buffer need to be consistent with the scaling factors and original PS buffer size.



**Figure 38-21. PS Buffer Scaling**

When sourcing a subset of the PS image, it should fall completely within the PS buffer to avoid displaying incorrect data. The following conditions should be met:

$$x\_base\_addr\_offset + x\_scale * (LRC\_X - ULC\_X) \leq PS\_pitch$$

$$y\_base\_addr\_offset + y\_scale * (LRC\_Y - ULC\_Y) \leq PS\_size$$

The PXP hardware does not check for these conditions and will render the image as programmed. The following case could indicate invalid programming parameters for the PXP:

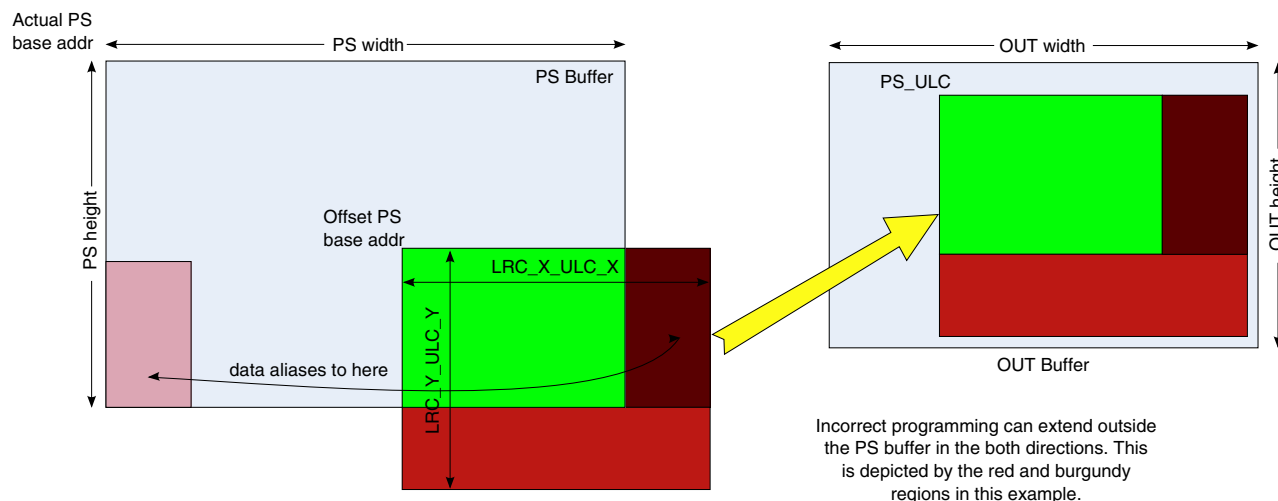


Figure 38-22. Example with Invalid Parameters

### 38.3.38 Color Key Processing

Pixels may be made transparent to the corresponding AS by using the PS color key registers.

If a PS pixel matches the range specified by the REG\_PS\_COLORKEYLOW and REG\_PS\_COLORKEYHIGH registers, the pixel from the associated AS will be displayed. If no AS is present for the pixel, a black pixel will be generated since the default AS pixel is 0x00000000 (transparent black pixel).

The most common use for this is when a bitmap does not support an alpha-field or for applications such as "green screen" where an image is substituted for a solid background color .





**Figure 38-23. The PS image (player) and AS image (stadium)**

The green portion of the background image can be color keyed to display the contents of the AS buffer for locations that match the color range. For this example, the color range is:

PS Colorkey:  $00 < R < 80$   $70 < G < ff$   $00 < B < 80$

The resulting image becomes:



**Figure 38-24. Resulting Image**

### 38.3.39 In Place Processing (PS buffer is destination buffer)

The PXP also has the ability to process an image and write the resulting buffer back to the original PS buffer. This is referred to as "in place" rendering.

This could be useful for basic blit operations into the PS buffer. IN\_PLACE operations are achieved by programming the OUT base address to the pixel location in the PS buffer that marks the upper left pixel of the update region. The actual region that is updated should be indicated by programming the ULC = (0,0) and the LRC = (X,Y). The region bounded by the coordinates will be updated, and the rest of the PS buffer will not be modified.

### 38.3.40 Alpha Surface (AS) Processing

The AS surface has a complete set of registers that determines how the AS effects the final OUT surface.

Most of the registers that exist for the PS surface also are defined for the AS surface where applicable. This is provided to replicate the SW interface for each PS and AS processes.

### 38.3.41 Alpha Handling

Alpha values in the AS are embedded in the source image pixels. For AS pixel formats that do not support an alpha value, the pixel is assigned an alpha value of 0xFF (opaque).

This can be modified by the AS control by setting either the ALPHA\_MULTIPLY or ALPHA\_OVERRIDE bit in the associated AS\_CTRL register. If ALPHA\_MULTIPLY is enabled, the 8-bit ALPHA value from the AS\_CTRL register is multiplied by the source alpha before blending with the PS image. If the ALPHA\_OVERRIDE bit is set, the 8-bit ALPHA value is simply substituted for the pixel.

### 38.3.42 Color Key Processing (AS\_CTRL)

The AS\_CTRL register also contains an ENABLE\_COLORKEY bit that can be used to enable or disable color key substitution for the AS.

When enabled, the pixel values are compared to the ASCOLORKEYLOW and ASCOLORKEYHIGH registers to determine if a match has occurred. When an AS pixel matches the color key range, the pixel from the AS image is considered transparent and the corresponding PS pixel is rendered. If both the PS and AS pixels match their corresponding color key ranges, the AS pixel is displayed unmodified.

AS color keys are handled in a manner similar to PS color keys. The same images used in the PS color key example could be used with the images swapped. In this case, matches on the AS image to the ASCOLORKEY register would display the PS pixels.

## 38.4 Output Image Processing

Several PXP options affect the resulting output image.

### 38.4.1 Output Image Size

The PXP generates an output image in the resolution programmed by the REG\_OUT\_LRC. As the PXP processes pixels, it iterates over the NxN blocks (in output scan-block order) based on the final image resolution.

### 38.4.2 Output Format

The result of PXP operations are written to the buffer pointed to by the REG\_OUT\_BUF/REG\_OUT\_BUF2 registers. The pixel format is controlled by the REG\_OUT\_CTRL[FORMAT] bit-field.

32-bit pixels are formed directly from the internal 24-bit representations and 16-bit pixel formats are generated by truncating the internal 24-bit values to the appropriate number of bits. For formats supporting an alpha value, the PXP assigns the alpha using the 8-bit value in the REG\_OUT\_CTRL[ALPHA] field. For ARGB1555, the most significant alpha bit is appended to the output pixel. Also, for ARGB4444, the most significant nibble is appended to the output pixel. Single and dual buffer YUV output formats are also available. Since each pixel in the data path is represented by a full YUV444 24bpp value, decimation reduces the output in cases of YUV422/420 output formats.

### 38.4.3 Rotation/Flip operations

The PXP supports four rotation angles in conjunction with vertical and horizontal flip options. The flip operations effectively take place before the rotation.

Rotations of 0, 90, 180, and 270 degrees are supported and any combination of rotation and flip are supported. There is no performance difference between any of these modes of operation.

## 38.5 Queuing PXP transactions

The PXP supports a primitive ability to queue up one operation while the current operation is running. This is enabled through the use of the REG\_NEXT register.

When this register is written, it enables the PXP to reload its current register contents with the data found at the location pointed to by this address when it completes processing of the current frame. This feature may be useful in helping to reduce the interrupt latency in servicing the PXP, especially in cases where the PXP and LCDIF are using the on-chip SRAM buffer handshake (since the PXP must begin generating next frame data immediately).

If the PXP is idle when the REG\_NEXT register is written, the PXP treats this as an indication that it should immediately load the values at the pointer and begin processing the frame. This ability should allow software to use the same routines when programming the PXP (so that the first frame doesn't differ from subsequent frames).

When loading values from the NEXT register, all registers in the PXP are reloaded. Some register loads have no effect

After writing the REG\_NEXT register, the PXP will set the REG\_NEXT[ENABLED] bit of the REG\_NEXT register to indicate that the next command has been queued. Software should first check the status of this bit to ensure that a previous command has not been enabled. Likewise, after programming the first frame in a sequence of frames, software should poll this bit until it is sampled logic 1'b0 before queuing the next operation.

The PXP will issue interrupts from frames as they complete, regardless of whether they were started by writing the control registers directly or using the REG\_NEXT register. When software receives an interrupt, it should check/clear the PXP's status register as normal, poll the REG\_PXP[ENABLED] bit, and then issue the next operation. A queued operation may be cancelled by issuing a CLEAR operation to the REG\_PXP[ENABLED] register bit. The SET and TOGGLE operations should never be used with this register.

## 38.6 Error Handling

The PXP does minimal checking on the control registers, so it is important that these are correctly specified. The PXP does monitor the bus transactions for errors and will report errors in the status register.

Upon receipt of a bus error, the PXP will set the ERROR interrupt and abort any further operations. Bus errors can be generated from any system access that results in an error response returned from the internal SIM Bus errors in the PXP are signaled as either a read or a write error, but do not indicate the failing address. Software may deduce the failing address from the current block status indicators.

### 38.6.1 Known PXP Limitations/Issues

The PXP has the following known limitations:

1. When using the NEXT register, the interrupt enable setting should remain the same for all frames. If not, the PXP will change the interrupt enable register value and possible cause the loss of an interrupt.
2. Rotations of 180/270 are not supported when performing LCD handshakes

## 38.7 PXP Memory Map/Register Definition

PXP Hardware Register Format Summary

**PXP memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20F_0000	Control Register 0 (PXP_CTRL)	32	R/W	C000_0000h	<a href="#">38.7.1/2316</a>
20F_0010	Status Register (PXP_STAT)	32	R/W	0000_0000h	<a href="#">38.7.2/2318</a>
20F_0020	Output Buffer Control Register (PXP_OUT_CTRL)	32	R/W	0000_0000h	<a href="#">38.7.3/2320</a>
20F_0030	Output Frame Buffer Pointer (PXP_OUT_BUF)	32	R/W	0000_0000h	<a href="#">38.7.4/2322</a>
20F_0040	Output Frame Buffer Pointer #2 (PXP_OUT_BUF2)	32	R/W	0000_0000h	<a href="#">38.7.5/2323</a>
20F_0050	Output Buffer Pitch (PXP_OUT_PITCH)	32	R/W	0000_0000h	<a href="#">38.7.6/2323</a>
20F_0060	Output Surface Lower Right Coordinate (PXP_OUT_LRC)	32	R/W	0000_0000h	<a href="#">38.7.7/2324</a>
20F_0070	Processed Surface Upper Left Coordinate (PXP_OUT_PS_ULC)	32	R/W	0000_0000h	<a href="#">38.7.8/2325</a>

*Table continues on the next page...*

## PXP memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20F_0080	Processed Surface Lower Right Coordinate (PXP_OUT_PS_LRC)	32	R/W	0000_0000h	<a href="#">38.7.9/2326</a>
20F_0090	Alpha Surface Upper Left Coordinate (PXP_OUT_AS_ULC)	32	R/W	0000_0000h	<a href="#">38.7.10/2327</a>
20F_00A0	Alpha Surface Lower Right Coordinate (PXP_OUT_AS_LRC)	32	R/W	0000_0000h	<a href="#">38.7.11/2328</a>
20F_00B0	Processed Surface (PS) Control Register (PXP_PS_CTRL)	32	R/W	0000_0000h	<a href="#">38.7.12/2329</a>
20F_00C0	PS Input Buffer Address (PXP_PS_BUF)	32	R/W	0000_0000h	<a href="#">38.7.13/2331</a>
20F_00D0	PS U/Cb or 2 Plane UV Input Buffer Address (PXP_PS_UBUF)	32	R/W	0000_0000h	<a href="#">38.7.14/2331</a>
20F_00E0	PS V/Cr Input Buffer Address (PXP_PS_VBUF)	32	R/W	0000_0000h	<a href="#">38.7.15/2332</a>
20F_00F0	Processed Surface Pitch (PXP_PS_PITCH)	32	R/W	0000_0000h	<a href="#">38.7.16/2333</a>
20F_0100	PS Background Color (PXP_PS_BACKGROUND)	32	R/W	0000_0000h	<a href="#">38.7.17/2334</a>
20F_0110	PS Scale Factor Register (PXP_PS_SCALE)	32	R/W	1000_1000h	<a href="#">38.7.18/2334</a>
20F_0120	PS Scale Offset Register (PXP_PS_OFFSET)	32	R/W	0000_0000h	<a href="#">38.7.19/2336</a>
20F_0130	PS Color Key Low (PXP_PS_CLRKEYLOW)	32	R/W	00FF_FFFFh	<a href="#">38.7.20/2336</a>
20F_0140	PS Color Key High (PXP_PS_CLRKEYHIGH)	32	R/W	0000_0000h	<a href="#">38.7.21/2337</a>
20F_0150	Alpha Surface Control (PXP_AS_CTRL)	32	R/W	0000_0000h	<a href="#">38.7.22/2338</a>
20F_0160	Alpha Surface Buffer Pointer (PXP_AS_BUF)	32	R/W	0000_0000h	<a href="#">38.7.23/2340</a>
20F_0170	Alpha Surface Pitch (PXP_AS_PITCH)	32	R/W	0000_0000h	<a href="#">38.7.24/2341</a>
20F_0180	Overlay Color Key Low (PXP_AS_CLRKEYLOW)	32	R/W	00FF_FFFFh	<a href="#">38.7.25/2341</a>
20F_0190	Overlay Color Key High (PXP_AS_CLRKEYHIGH)	32	R/W	0000_0000h	<a href="#">38.7.26/2342</a>
20F_01A0	Color Space Conversion Coefficient Register 0 (PXP_CSC1_COEF0)	32	R/W	0400_0000h	<a href="#">38.7.27/2343</a>
20F_01B0	Color Space Conversion Coefficient Register 1 (PXP_CSC1_COEF1)	32	R/W	0123_0208h	<a href="#">38.7.28/2344</a>
20F_01C0	Color Space Conversion Coefficient Register 2 (PXP_CSC1_COEF2)	32	R/W	079B_076Ch	<a href="#">38.7.29/2345</a>
20F_01D0	Color Space Conversion Control Register. (PXP_CSC2_CTRL)	32	R/W	0000_0001h	<a href="#">38.7.30/2346</a>

Table continues on the next page...

## PXP memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20F_01E0	Color Space Conversion Coefficient Register 0 (PXP_CSC2_COEF0)	32	R/W	0000_0000h	<a href="#">38.7.31/2347</a>
20F_01F0	Color Space Conversion Coefficient Register 1 (PXP_CSC2_COEF1)	32	R/W	0000_0000h	<a href="#">38.7.32/2348</a>
20F_0200	Color Space Conversion Coefficient Register 2 (PXP_CSC2_COEF2)	32	R/W	0000_0000h	<a href="#">38.7.33/2348</a>
20F_0210	Color Space Conversion Coefficient Register 3 (PXP_CSC2_COEF3)	32	R/W	0000_0000h	<a href="#">38.7.34/2349</a>
20F_0220	Color Space Conversion Coefficient Register 4 (PXP_CSC2_COEF4)	32	R/W	0000_0000h	<a href="#">38.7.35/2350</a>
20F_0230	Color Space Conversion Coefficient Register 5 (PXP_CSC2_COEF5)	32	R/W	0000_0000h	<a href="#">38.7.36/2350</a>
20F_0240	Lookup Table Control Register. (PXP_LUT_CTRL)	32	R/W	8001_0000h	<a href="#">38.7.37/2351</a>
20F_0250	Lookup Table Control Register. (PXP_LUT_ADDR)	32	R/W	0000_0000h	<a href="#">38.7.38/2353</a>
20F_0260	Lookup Table Data Register. (PXP_LUT_DATA)	32	R/W	0000_0000h	<a href="#">38.7.39/2355</a>
20F_0270	Lookup Table External Memory Address Register. (PXP_LUT_EXTMEM)	32	R/W	0000_0000h	<a href="#">38.7.40/2355</a>
20F_0280	Color Filter Array Register. (PXP_CFA)	32	R/W	0000_0000h	<a href="#">38.7.41/2356</a>
20F_0290	Histogram Control Register. (PXP_HIST_CTRL)	32	R/W	0000_0020h	<a href="#">38.7.42/2356</a>
20F_02A0	2-level Histogram Parameter Register. (PXP_HIST2_PARAM)	32	R/W	0000_0F00h	<a href="#">38.7.43/2357</a>
20F_02B0	4-level Histogram Parameter Register. (PXP_HIST4_PARAM)	32	R/W	0F0A_0500h	<a href="#">38.7.44/2358</a>
20F_02C0	8-level Histogram Parameter 0 Register. (PXP_HIST8_PARAM0)	32	R/W	0604_4000h	<a href="#">38.7.45/2359</a>
20F_02D0	8-level Histogram Parameter 1 Register. (PXP_HIST8_PARAM1)	32	R/W	0F0D_0B09h	<a href="#">38.7.46/2360</a>
20F_02E0	16-level Histogram Parameter 0 Register. (PXP_HIST16_PARAM0)	32	R/W	0302_0100h	<a href="#">38.7.47/2361</a>
20F_02F0	16-level Histogram Parameter 1 Register. (PXP_HIST16_PARAM1)	32	R/W	0706_0504h	<a href="#">38.7.48/2362</a>
20F_0300	16-level Histogram Parameter 2 Register. (PXP_HIST16_PARAM2)	32	R/W	0B0A_0908h	<a href="#">38.7.49/2363</a>
20F_0310	16-level Histogram Parameter 3 Register. (PXP_HIST16_PARAM3)	32	R/W	0F0E_0D0Ch	<a href="#">38.7.50/2364</a>
20F_0320	PXP Power Control Register. (PXP_POWER)	32	R/W	0000_0000h	<a href="#">38.7.51/2365</a>
20F_0400	Next Frame Pointer (PXP_NEXT)	32	R/W	0000_0000h	<a href="#">38.7.52/2366</a>

### 38.7.1 Control Register 0 (PXP\_CTRL)

The CTRL register contains controls for the PXP module.

PXP\_CTRL: 0x000

PXP\_CTRL\_SET: 0x004

PXP\_CTRL\_CLR: 0x008

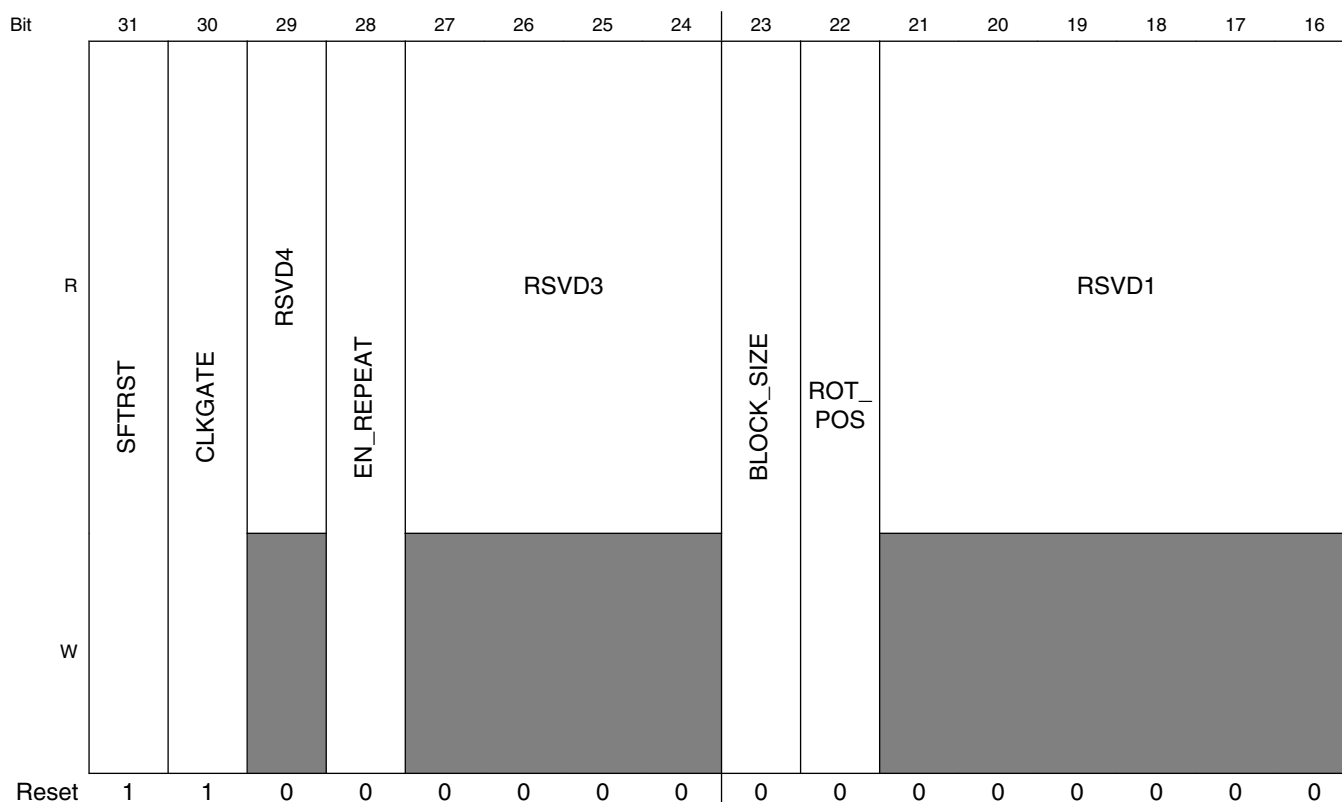
PXP\_CTRL\_TOG: 0x00C

The Control register contains the primary controls for the PXP block. The present bits indicate which of the sub-features of the block are present in the hardware.

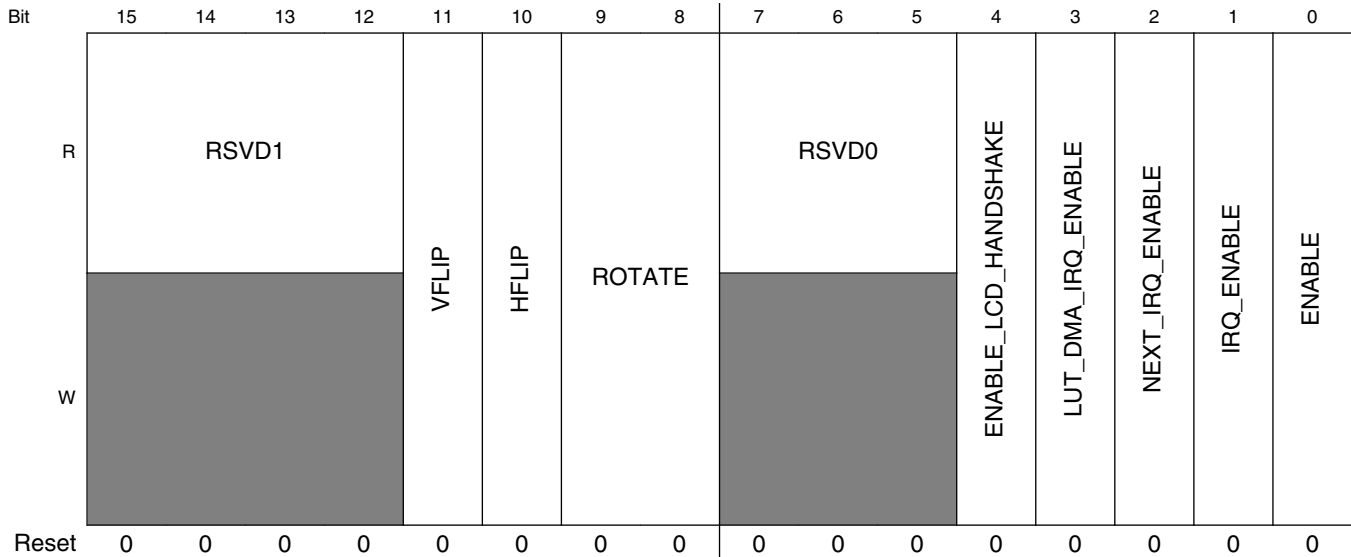
#### EXAMPLE

```
PXP_CTRL_SET(BM_PXP_CTRL_SFTRST);
PXP_CTRL_CLR(BM_PXP_CTRL_SFTRST | BM_PXP_CTRL_CLKGATE);
```

Address: 20F\_0000h base + 0h offset = 20F\_0000h







PXP\_CTRL field descriptions

Field	Description
31 SFTRST	Set this bit to zero to enable normal PXP operation. Set this bit to one (default) to disable clocking with the PXP and hold it in its reset (lowest power) state. This bit can be turned on and then off to reset the PXP block to its default state.
30 CLKGATE	This bit must be set to zero for normal operation. When set to one it gates off the clocks to the block.
29 RSVD4	Reserved, always set to zero.
28 EN_REPEAT	Enable the PXP to run continuously. When this bit is set, the PXP will repeat based on the current configuration register settings. If this bit is not set, the PXP will complete the process and enter the idle state ready to accept the next frame to be processed. This bit should be set when the LCDIF handshake mode is enabled so that the next frame is automatically generated for the next screen refresh cycle. If it not set and the handshake mode is enabled, the CPU will have to initiate the PXP for the next refresh cycle. When the PXP NEXT feature is used, it has priority over the REPEAT mode, in that the new register settings are fetched first, and then the next PXP operation will continue.
27–24 RSVD3	Reserved, always set to zero.
23 BLOCK_SIZE	Select the block size to process. 0x0 <b>8X8</b> — Process 8x8 pixel blocks. 0x1 <b>16X16</b> — Process 16x16 pixel blocks.
22 ROT_POS	This bit controls where rotation will occur in the PXP datapath. Setting this bit to 1'b0 will place the rotation resources at the output stage of the PXP data path. Image compositing will occur before pixels are processed for rotation. Setting this bit to a 1'b1 will place the rotation resources before image composition. Only the PS can be rotated in this configuration and AS will not be rotated.
21–12 RSVD1	Reserved, always set to zero.
11 VFLIP	Indicates that the output buffer should be flipped vertically (effect applied before rotation).
10 HFLIP	Indicates that the output buffer should be flipped horizontally (effect applied before rotation).

Table continues on the next page...

**PXP\_CTRL field descriptions (continued)**

Field	Description
9–8 ROTATE	Indicates the clockwise rotation to be applied at the output buffer. The rotation effect is defined as occurring after the FLIP_X and FLIP_Y permutation.  0x0 <b>ROT_0</b> — 0x1 <b>ROT_90</b> — 0x2 <b>ROT_180</b> — 0x3 <b>ROT_270</b> —
7–5 RSVD0	Reserved, always set to zero.
4 ENABLE_LCD_HANDSHAKE	Enable handshake with LCD controller. When this is set, the PXP will not process an entire framebuffer, but will instead process rows of NxN blocks in a double-buffer handshake with the LCDIF. This enables the use of the onboard SRAM for a partial frame buffer.
3 LUT_DMA_IRQ_ENABLE	LUT DMA interrupt enable. When set, the PXP will issue an interrupt when the LUT DMA has finished transferring data.
2 NEXT_IRQ_ENABLE	Next command interrupt enable. When set, the PXP will issue an interrupt when a queued command initiated by a write to the PXP_NEXT register has been loaded into the PXP's registers. This interrupt also indicates that a new command may now be queued.
1 IRQ_ENABLE	Interrupt enable. NOTE: When using the PXP_NEXT functionality to reprogram the PXP, the new value of this bit will be used and may therefore enable or disable an interrupt unintentionally.
0 ENABLE	Enables PXP operation with specified parameters. The ENABLE bit will remain set while the PXP is active and will be cleared once the current operation completes. Software should use the IRQ bit in the PXP_STAT when polling for PXP completion.

**38.7.2 Status Register (PXP\_STAT)**

The PXP Interrupt Status register provides interrupt status information.

PXP\_STAT: 0x010

PXP\_STAT\_SET: 0x014

PXP\_STAT\_CLR: 0x018

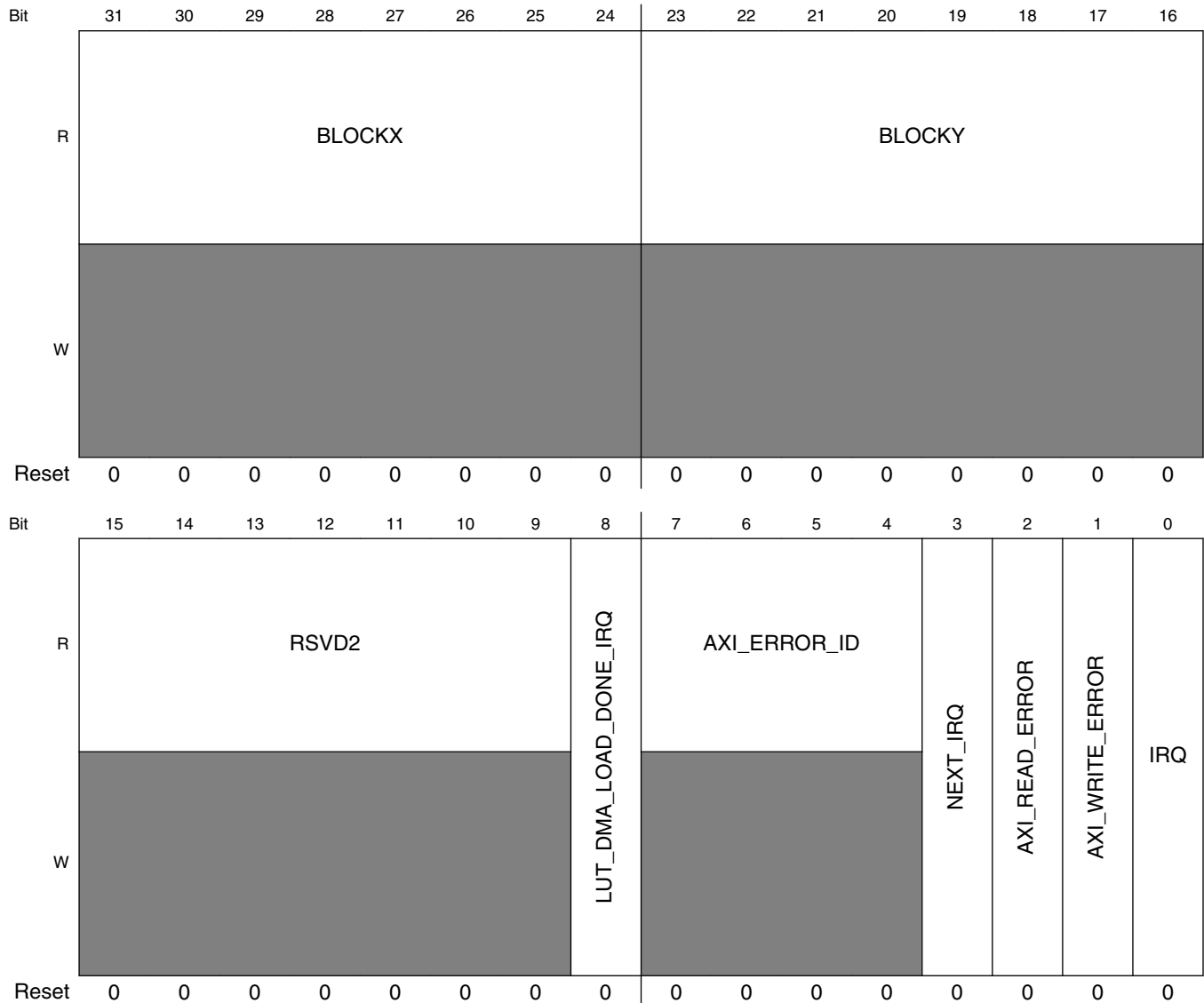
PXP\_STAT\_TOG: 0x01C

This register provides PXP interrupt status and the current X/Y block coordinate that is being processed.

**EXAMPLE**

```
PXP_STAT_CLR(BM_PXP_STAT_IRQ); // clear CSC interrupt
```

Address: 20F\_0000h base + 10h offset = 20F\_0010h

**PXP\_STAT field descriptions**

Field	Description
31–24 BLOCKX	Indicates the X coordinate of the block currently being rendered.
23–16 BLOCKY	Indicates the X coordinate of the block currently being rendered.
15–9 RSVD2	Reserved, always set to zero.
8 LUT_DMA_LOAD_DONE_IRQ	Indicates that the LUT DMA transfer has completed.
7–4 AXI_ERROR_ID	Indicates the AXI ID of the failing bus operation.

*Table continues on the next page...*

**PXP\_STAT field descriptions (continued)**

Field	Description
3 NEXT_IRQ	Indicates that a command issued with the "Next Command" functionality has been issued and that a new command may be initiated with a write to the PXP_NEXT register.
2 AXI_READ_ERROR	Indicates PXP encountered an AXI read error and processing has been terminated.
1 AXI_WRITE_ERROR	Indicates PXP encountered an AXI write error and processing has been terminated.
0 IRQ	Indicates current PXP interrupt status. The IRQ is routed through the pxp_irq when the IRQ_ENABLE bit in the control register is set.

**38.7.3 Output Buffer Control Register (PXP\_OUT\_CTRL)**

The OUT\_CTRL register contains controls for the Output Buffer.

PXP\_OUT\_CTRL: 0x020

PXP\_OUT\_CTRL\_SET: 0x024

PXP\_OUT\_CTRL\_CLR: 0x028

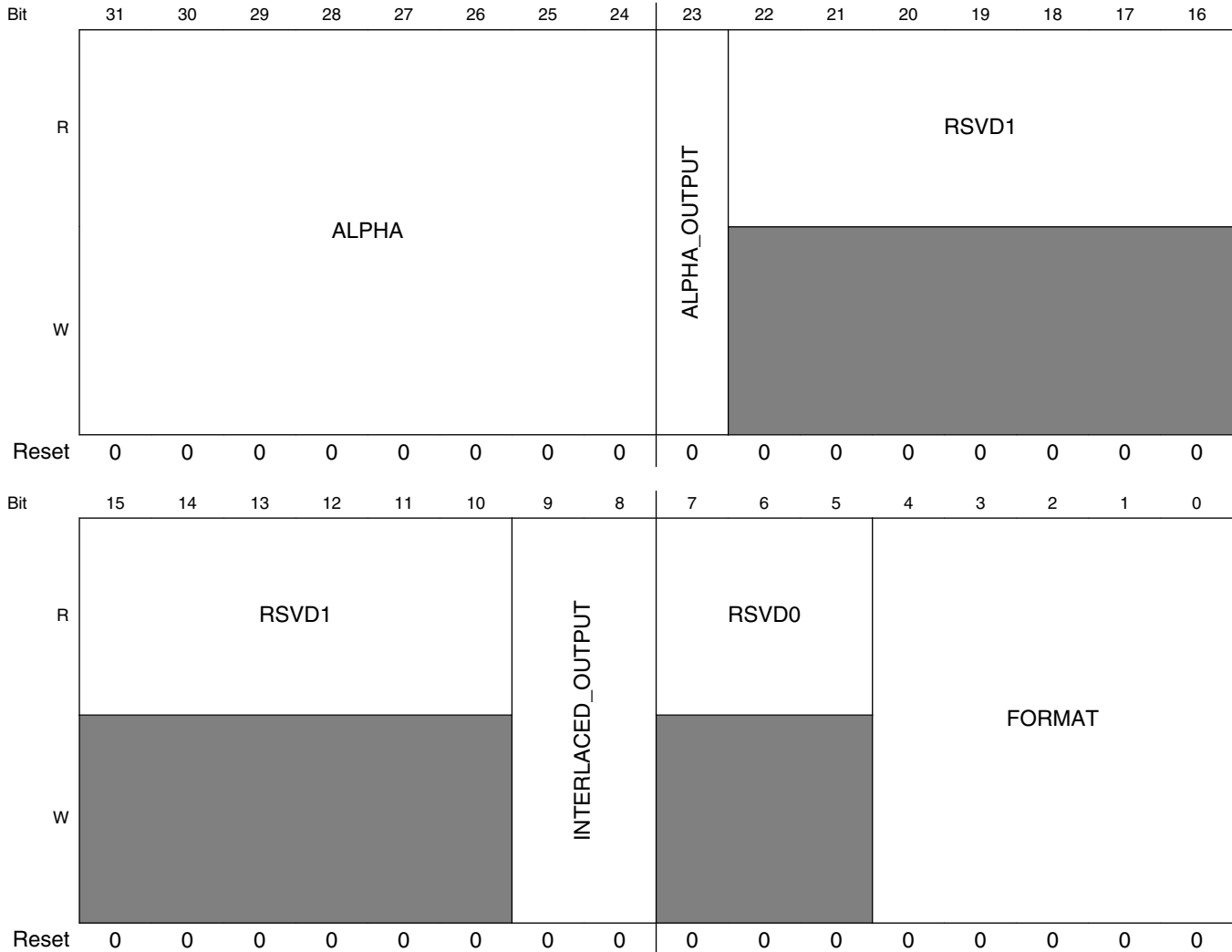
PXP\_OUT\_CTRL\_TOG: 0x02C

The Control register contains the primary controls for the PXP block. The present bits indicate which of the sub-features of the block are present in the hardware.

**EXAMPLE**

```
PXP_CTRL_SET(BM_PXP_CTRL_SFTRST);
PXP_CTRL_CLR(BM_PXP_CTRL_SFTRST | BM_PXP_CTRL_CLKGATE);
```

Address: 20F\_0000h base + 20h offset = 20F\_0020h

**PXP\_OUT\_CTRL field descriptions**

Field	Description
31–24 ALPHA	When generating an output buffer with an alpha component, the value in this field will be used when enabled to override the alpha passed through the pixel data pipeline.
23 ALPHA_OUTPUT	Indicates that alpha component in output buffer pixels should be overwritten by PXP_OUT_CTRL[ALPHA]. If 0, retain their alpha value from the computed alpha for that pixel.
22–10 RSVD1	Reserved, always set to zero.
9–8 INTERLACED_OUTPUT	Determines how the PXP writes it's output data. Output interlacing should not be used in conjunction with input interlacing. Splitting frames into fields is most efficient using output interlacing. 2-plane output formats AND interlaced output is NOT supported.  0x0 <b>PROGRESSIVE</b> — All data written in progressive format to the OUTBUF Pointer. 0x1 <b>FIELD0</b> — Interlaced output: only data for field 0 is written to the OUTBUF Pointer.

*Table continues on the next page...*

## PXP\_OUT\_CTRL field descriptions (continued)

Field	Description
0x2	<b>FIELD1</b> — Interlaced output: only data for field 1 is written to the OUTBUF2 Pointer.
0x3	<b>INTERLACED</b> — Interlaced output: data for field 0 is written to OUTBUF and data for field 1 is written to OUTBUF2.
7–5 RSVD0	Reserved, always set to zero.
4–0 FORMAT	Output framebuffer format. The UV byte lanes are synonymous with CbCr byte lanes for YUV output pixel formats. For example, the YUV2P420 format should be selected when the output is YCbCr 2-plane 420 output format.  0x0 <b>ARGB8888</b> — 32-bit pixels 0x4 <b>RGB888</b> — 32-bit pixels (unpacked 24-bit pixel in 32 bit DWORD.) 0x5 <b>RGB888P</b> — 24-bit pixels (packed 24-bit format) 0x8 <b>ARGB1555</b> — 16-bit pixels 0x9 <b>ARGB4444</b> — 16-bit pixels 0xC <b>RGB555</b> — 16-bit pixels 0xD <b>RGB444</b> — 16-bit pixels 0xE <b>RGB565</b> — 16-bit pixels 0x10 <b>YUV1P444</b> — 32-bit pixels (1-plane XYUV unpacked) 0x12 <b>UYVY1P422</b> — 16-bit pixels (1-plane U0,Y0,V0,Y1 interleaved bytes) 0x13 <b>VYUY1P422</b> — 16-bit pixels (1-plane V0,Y0,U0,Y1 interleaved bytes) 0x14 <b>Y8</b> — 8-bit monochrome pixels (1-plane Y luma output) 0x15 <b>Y4</b> — 4-bit monochrome pixels (1-plane Y luma, 4 bit truncation) 0x18 <b>YUV2P422</b> — 16-bit pixels (2-plane UV interleaved bytes) 0x19 <b>YUV2P420</b> — 16-bit pixels (2-plane UV) 0x1A <b>YVU2P422</b> — 16-bit pixels (2-plane VU interleaved bytes) 0x1B <b>YVU2P420</b> — 16-bit pixels (2-plane VU)

## 38.7.4 Output Frame Buffer Pointer (PXP\_OUT\_BUF)

Output Framebuffer Pointer. This register points to the beginning of the output frame buffer. This pointer is used for progressive format and field 0 when generating interlaced output.

This register is used by the logic to point to the current output location for the output frame buffer.

## EXAMPLE

```
PXP_OUT_BUF_WR( buffer );
```

Address: 20F\_0000h base + 30h offset = 20F\_0030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**PXP\_OUT\_BUF field descriptions**

Field	Description
31–0 ADDR	Current address pointer for the output frame buffer. The address can have any byte alignment. 64B alignment is recommended for optimal performance.

**38.7.5 Output Frame Buffer Pointer #2 (PXP\_OUT\_BUF2)**

Output Framebuffer Pointer #2. This register points to the beginning of the output frame buffer for either field 1 when generating interlaced output or for the UV buffer when in YUV 2-plane output modes. Both interlaced output AND 2-plane output modes are not supported in a single PXP operation. This register is NOT used as the pointer to the 2nd buffer when in LCDIF\_HANDSHAKE mode.

This register is used by the logic to point to the current output location for the field 1 or UV output frame buffer.

**EXAMPLE**

```
PXP_OUT_BUF_WR( field0 ); // buffer for interlaced field 0
PXP_OUT_BUF2_WR( field1 ); // buffer for interlaced field 1
```

Address: 20F\_0000h base + 40h offset = 20F\_0040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**PXP\_OUT\_BUF2 field descriptions**

Field	Description
31–0 ADDR	Current address pointer for the output frame buffer. The address can have any byte alignment. 64B alignment is recommended for optimal performance.

**38.7.6 Output Buffer Pitch (PXP\_OUT\_PITCH)**

This register contains the output buffer pitch in bytes.

Any byte value will indicate the vertical pitch. This value will be used in output pixel address calculations.

**EXAMPLE**

## PXP Memory Map/Register Definition

```
PXP_OUT_PITCH_WR( 68 * 4 ); // The output buffer pitch is 68 pixels times 32 bits per pixel
```

Address: 20F\_0000h base + 50h offset = 20F\_0050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD																PITCH															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### PXP\_OUT\_PITCH field descriptions

Field	Description
31–16 RSVD	Reserved, always set to zero.
15–0 PITCH	Indicates the number of bytes in memory between two vertically adjacent pixels.

## 38.7.7 Output Surface Lower Right Coordinate (PXP\_OUT\_LRC)

This register contains the size, or lower right coordinate, of the output buffer NOT rotated. It is implied that the upper left coordinate of the output surface is always [0,0]. When rotating the framebuffer, the PXP will automatically swap the X/Y, or WIDTH/HEIGHT, to accommodate the rotated size.

This register sets the size of the output frame buffer in pixels, not blocks. The frame buffer need not be a multiple of NxN pixels. Partial blocks will be written for output frame buffer sizes that are not divisible by N pixels in either dimension.

### EXAMPLE

```
PXP_OUT_LRC[X]=319; // set width of output frame buffer to 320 pixels
PXP_OUT_LRC[Y]=243; // set height of output frame buffer to 244 pixels which is not
divisible by block size N
```

```
PXP_OUT_LRC_WR( BF_PXP_OUT_LRC_X(319) | BF_PXP_OUT_LRC_Y(243) );
```

Address: 20F\_0000h base + 60h offset = 20F\_0060h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RSVD1								X							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD0								Y							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



**PXP\_OUT\_LRC field descriptions**

Field	Description
31–30 RSVD1	Reserved, always set to zero.
29–16 X	Indicates number of horizontal PIXELS in the output surface (non-rotated). The output buffer pixel width minus 1 should be programmed. The image size is not required to be a multiple of 8 pixels. The PXP will clip the pixel output at this boundary.
15–14 RSVD0	Reserved, always set to zero.
13–0 Y	Indicates the number of vertical PIXELS in the output surface (non-rotated). The output buffer pixel height minus 1 should be programmed. The image size is not required to be a multiple of 8 pixels. The PXP will clip the pixel output at this boundary.

### 38.7.8 Processed Surface Upper Left Coordinate (PXP\_OUT\_PS\_ULC)

This register contains the upper left pixel coordinate for the Processed Surface in the OUTPUT buffer.

This register contains the upper left coordinate of the Processed Surface in the output frame buffer (in pixels). Values that are within the PXP\_OUT\_LRC X,Y extents are valid. The lowest valid value for these fields is 0,0. If the value of the PXP\_OUT\_PS\_ULC is greater than the PXP\_OUT\_LRC, then no PS pixels will be fetched from memory, but only PXP\_PS\_BACKGROUND pixels will be processed by the PS engine. Pixel locations that are greater than or equal to the PS upper left coordinates, less than or equal to the PS lower right coordinates, and within the PXP\_OUT\_LRC extents will use the PS to render pixels into the output buffer.

#### EXAMPLE

```
PXP_OUT_PS_ULC_WR(0,0x0002_0002); // Processed Surface upper left coordinate at (X,Y) = 2,2. The PS surface will not effect pixels in the first and second row and column of the output buffer.
```

Address: 20F\_0000h base + 70h offset = 20F\_0070h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RSVD1		X													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD0		Y													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**PXP\_OUT\_PS\_ULC field descriptions**

Field	Description
31–30 RSVD1	Reserved, always set to zero.
29–16 X	This field indicates the upper left X-coordinate (in pixels) of the processed surface (PS) in the output buffer.
15–14 RSVD0	Reserved, always set to zero.
13–0 Y	This field indicates the upper left Y-coordinate (in pixels) of the processed surface in the output buffer.

### 38.7.9 Processed Surface Lower Right Coordinate (PXP\_OUT\_PS\_LRC)

This register contains the lower right extent for the Processed Surface in the OUTPUT buffer.

This register contains the lower right coordinate of the Processed Surface in the output frame buffer (in pixels). Values that are within the PXP\_OUT\_LRC X,Y extents are valid. The lowest valid value for these fields is 0,0. Pixel locations that are greater than or equal to the PS upper left coordinates, less than or equal to the PS lower right coordinates, and within the PXP\_OUT\_LRC extents will use the PS to render pixels into the output buffer.

#### EXAMPLE

```
PXP_OUT_PS_ULC_WR(0,0x03FF_03FF); // With this UL/LR pair of pixel coordinates, only one
pixel at OUT[X,Y]=1023,1023 will use the PS to contribute to its value.
PXP_OUT_PS_LRC_WR(0,0x03FF_03FF);
```

Address: 20F\_0000h base + 80h offset = 20F\_0080h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RSVD1		X													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD0		Y													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**PXP\_OUT\_PS\_LRC field descriptions**

Field	Description
31–30 RSVD1	Reserved, always set to zero.
29–16 X	This field indicates the lower right X-coordinate (in pixels) of the processed surface (PS) in the output frame buffer.
15–14 RSVD0	Reserved, always set to zero.
13–0 Y	This field indicates the lower right Y-coordinate (in pixels) of the processed surface in the output frame buffer.

**38.7.10 Alpha Surface Upper Left Coordinate (PXP\_OUT\_AS\_ULC)**

This register contains the upper left location for the Alpha Surface in the output buffer.

This register contains the upper left coordinate of AS in the output frame buffer (in pixels). Values that are within the PXP\_OUT\_LRC X,Y extents are valid. The lowest valid value for these fields is 0,0. Pixel locations that are greater than or equal to the upper left coordinates will use the AS to render pixels in the output buffer.

**EXAMPLE**

```
PXP_OUT_AS_ULC_WR(0,0x0001_0001); // Alpha Surface upper left coordinate at (X,Y) = 1,1.
The AS surface will not effect pixels in the first row or first column of the output buffer.
```

Address: 20F\_0000h base + 90h offset = 20F\_0090h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RSVD1		X													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD0		Y													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**PXP\_OUT\_AS\_ULC field descriptions**

Field	Description
31–30 RSVD1	Reserved, always set to zero.

*Table continues on the next page...*

**PXP\_OUT\_AS\_ULC field descriptions (continued)**

Field	Description
29–16 X	This field indicates the upper left X-coordinate (in pixels) of the alpha surface (AS) in the output frame buffer.
15–14 RSVD0	Reserved, always set to zero.
13–0 Y	This field indicates the upper left Y-coordinate (in pixels) of the alpha surface in the output frame buffer.

### 38.7.11 Alpha Surface Lower Right Coordinate (PXP\_OUT\_AS\_LRC)

This register contains the lower right extent for Alpha Surface in the output buffer.

This register contains the lower right coordinate of AS in the output frame buffer (in pixels). Values that are within the PXP\_OUT\_LRC X,Y extents are valid. The lowest valid value for these fields is 0,0. Pixel locations that are less than or equal to the lower right coordinates will use the AS to render pixels in the output buffer.

#### EXAMPLE

```
PXP_AS_LRC_WR(0,0x03FF_03FF); // Alpha Surface lower right coordinate at (X,Y) = 1023,1023.
```

Address: 20F\_0000h base + A0h offset = 20F\_00A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RSVD1		X													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD0		Y													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**PXP\_OUT\_AS\_LRC field descriptions**

Field	Description
31–30 RSVD1	Reserved, always set to zero.
29–16 X	This field indicates the lower right X-coordinate (in pixels) of the alpha surface (AS) in the output frame buffer.
15–14 RSVD0	Reserved, always set to zero.

*Table continues on the next page...*

**PXP\_OUT\_AS\_LRC field descriptions (continued)**

Field	Description
13–0 Y	This field indicates the lower right Y-coordinate (in pixels) of the alpha surface in the output frame buffer.

### 38.7.12 Processed Surface (PS) Control Register (PXP\_PS\_CTRL)

The PS\_CTRL register contains controls for the Processed Surface Buffer.

PXP\_PS\_CTRL: 0x0B0

PXP\_PS\_CTRL\_SET: 0x0b4

PXP\_PS\_CTRL\_CLR: 0x0B8

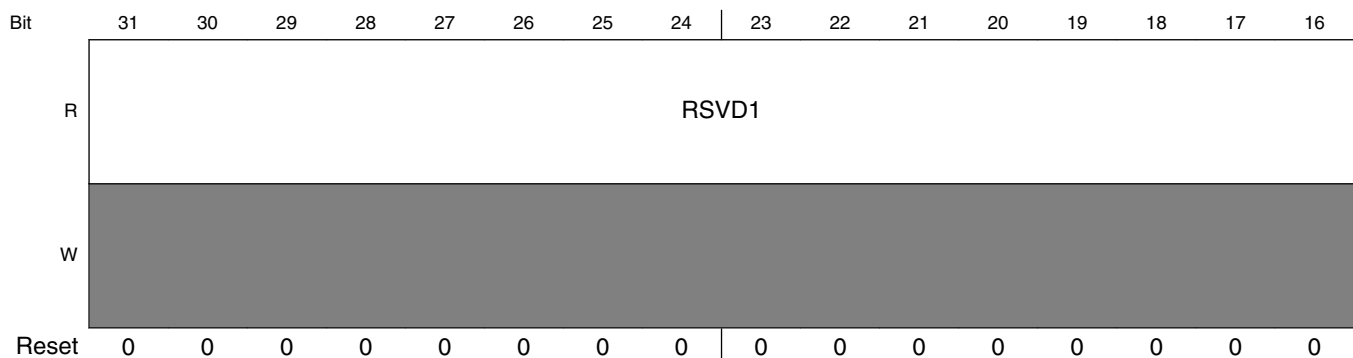
PXP\_PS\_CTRL\_TOG: 0x0BC

The Control register contains the primary controls for the PXP block. The present bits indicate which of the sub-features of the block are present in the hardware.

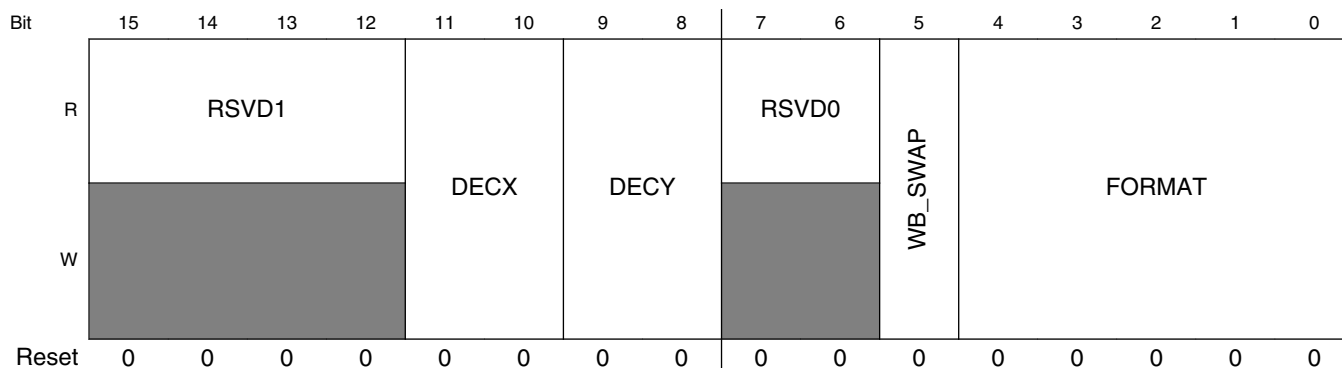
#### EXAMPLE

```
PXP_CTRL_SET(BM_PXP_CTRL_SFTRST);
PXP_CTRL_CLR(BM_PXP_CTRL_SFTRST | BM_PXP_CTRL_CLKGATE);
```

Address: 20F\_0000h base + B0h offset = 20F\_00B0h



## PXP Memory Map/Register Definition



### PXP\_PS\_CTRL field descriptions

Field	Description
31–12 RSVD1	Reserved, always set to zero.
11–10 DECX	Horizontal pre decimation filter control.  0x0 <b>DISABLE</b> — Disable pre-decimation filter. 0x1 <b>DECX2</b> — Decimate PS by 2. 0x2 <b>DECX4</b> — Decimate PS by 4. 0x3 <b>DECX8</b> — Decimate PS by 8.
9–8 DECY	Verticle pre decimation filter control.  0x0 <b>DISABLE</b> — Disable pre-decimation filter. 0x1 <b>DECY2</b> — Decimate PS by 2. 0x2 <b>DECY4</b> — Decimate PS by 4. 0x3 <b>DECY8</b> — Decimate PS by 8.
7–6 RSVD0	Reserved, always set to zero.
5 WB_SWAP	Swap bytes in words. For each 16 bit word, the two bytes will be swapped.
4–0 FORMAT	PS buffer format. To select between YUV and YCbCr formats, see bit 31 of the CSC1_COEF0 register.  0x4 <b>RGB888</b> — 32-bit pixels (unpacked 24-bit format) 0xC <b>RGB555</b> — 16-bit pixels 0xD <b>RGB444</b> — 16-bit pixels 0xE <b>RGB565</b> — 16-bit pixels 0x10 <b>YUV1P444</b> — 32-bit pixels (1-plane XYUV unpacked) 0x12 <b>UYVY1P422</b> — 16-bit pixels (1-plane U0,Y0,V0,Y1 interleaved bytes) 0x13 <b>VYUY1P422</b> — 16-bit pixels (1-plane V0,Y0,U0,Y1 interleaved bytes) 0x14 <b>Y8</b> — 8-bit monochrome pixels (1-plane Y luma output) 0x15 <b>Y4</b> — 4-bit monochrome pixels (1-plane Y luma, 4 bit truncation) 0x18 <b>YUV2P422</b> — 16-bit pixels (2-plane UV interleaved bytes) 0x19 <b>YUV2P420</b> — 16-bit pixels (2-plane UV) 0x1A <b>YVU2P422</b> — 16-bit pixels (2-plane VU interleaved bytes) 0x1B <b>YVU2P420</b> — 16-bit pixels (2-plane VU) 0x1E <b>YUV422</b> — 16-bit pixels (3-plane format) 0x1F <b>YUV420</b> — 16-bit pixels (3-plane format)

### 38.7.13 PS Input Buffer Address (PXP\_PS\_BUF)

PS Input Buffer Address. This should be programmed to the starting address of the RGB data or Y (luma) data for the PS plane.

This register contains the pointer to the Luma/RGB buffer. If the application requires an offset into the PS buffer, then this address can be set so that the desired offset is achieved. Any byte address is valid. For best performance, 64B alignment is recommended.

#### EXAMPLE

```
PXP_PS_BUF_WR(image_rgb); // RGB image
PXP_PS_BUF_WR(image_y);  // Y (luma) image data
PXP_PS_UBUF_WR(image_u); // U (Cb) image data
PXP_PS_VBUF_WR(image_v); // V (Cr) image data
```

Address: 20F\_0000h base + C0h offset = 20F\_00C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### PXP\_PS\_BUF field descriptions

Field	Description
31–0 ADDR	Address pointer for the PS RGB or Y (luma) input buffer.

### 38.7.14 PS U/Cb or 2 Plane UV Input Buffer Address (PXP\_PS\_UBUF)

PS Chroma (U/Cb/UV) Input Buffer Address. This register points to the beginning of the PS U/Cb input buffer. In two plane operation, this register points to the beginning of the PS UV chroma input buffer.

This register contains the pointer to the Chroma U/Cb or 2 plane UV buffer. This register is unused when processing 1-plane buffer formats. If the application requires an offset into the PS buffer, then this address can be set so that the desired offset is achieved. Any byte address is valid. For best performance, 64B alignment is recommended.

#### EXAMPLE

```
PXP_PS_BUF_WR(image_y); // Y (luma) image data
PXP_PS_UBUF_WR(image_u); // U (Cb) image data
```

## PXP Memory Map/Register Definition

```
PXP_PS_VBUF_WR(image_v); // V (Cr) image data
```

Address: 20F\_0000h base + D0h offset = 20F\_00D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### PXP\_PS\_UBUF field descriptions

Field	Description
31–0 ADDR	Address pointer for the PS U/Cb or 2 plane UV Chroma input buffer.

## 38.7.15 PS V/Cr Input Buffer Address (PXP\_PS\_VBUF)

PS Chroma (V/Cr) Input Buffer Address. This register points to the beginning of the PS V/Cr input buffer. In one or two plane operation, this register is not used. In monochrome modes Y8 and Y4, the low 16 bits are used as the U/V data in the datapath instead of sourcing U/V data from external buffers. In this case, it represents a fixed value for U/V data.

This register contains the pointer to the Chroma V/Cr buffer. For Y8/Y4 modes, the low 16 bits are used as the monochrome U and V values in the data path. Bits [15:8] represent the U data byte, and bits [7:0] represent the V data byte. Other than with Y8/Y4 input buffer formats, this register is unused when processing 1 or 2-plane buffer formats. If the application requires an offset into the PS buffer, then this address can be set so that the desired offset is achieved. Any byte address is valid. For best performance, 64B alignment is recommended.

### EXAMPLE

```
PXP_PS_BUF_WR(image_y); // Y (luma) image data
PXP_PS_UBUF_WR(image_u); // U (Cb) image data
PXP_PS_VBUF_WR(image_v); // V (Cr) image data
```

Address: 20F\_0000h base + E0h offset = 20F\_00E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	



**PXP\_PS\_VBUF field descriptions**

Field	Description
31–0 ADDR	Address pointer for the PS V/Cr Chroma input buffer.

**38.7.16 Processed Surface Pitch (PXP\_PS\_PITCH)**

This register contains the processed surface pitch in bytes.

Any byte value will indicate the vertical pitch of the PS source frame buffer. This value will be used in PS pixel address calculations. This value has no relation to the UL and LR registers. It specifies how many bytes are between two vertically adjacent pixels in the input PS surface. For multi-plane formats, the Y buffer pitch should be programmed. For 2-plane YUV422, the UV pitch is the same as the Y pitch. For 3-plane YUV422, the U and V pitch is 1/2 the Y pitch. For 2-plane YUV420, the UV pitch is 1/2 the Y pitch. For 3-plane YUV420, the U and V pitch is 1/4 the Y pitch. All source buffers should comply with these U and V resolution reductions with respect to their Y source buffers.

**EXAMPLE**

```
PXP_PS_PITCH_WR( 64 * 4 ); // The output buffer pitch is 64 pixels times 32 bits per pixel
```

Address: 20F\_0000h base + F0h offset = 20F\_00F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD																PITCH															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**PXP\_PS\_PITCH field descriptions**

Field	Description
31–16 RSVD	Reserved, always set to zero.
15–0 PITCH	Indicates the number of bytes in memory between two vertically adjacent pixels.

### 38.7.17 PS Background Color (PXP\_PS\_BACKGROUND)

PS Background Pixel Color. This register provides a pixel value used when processing pixels outside of the region specified by the PS Coordinate registers. This value can effectively be used to set the color of the letterboxing region around the PS image.

This register contains a pixel value to be used for any PS pixels that fall outside the PS extents. This is effectively a background or letterbox color. The CSC1 control and datapath pixel format should be considered when selecting the background color.

#### EXAMPLE

```
PXP_PS_BACKGROUND_WR(0x00000000); // letterbox is black
PXP_PS_BACKGROUND_WR(0x00800000); // letterbox is dark red
PXP_PS_BACKGROUND_WR(0x00008000); // letterbox is dark green
PXP_PS_BACKGROUND_WR(0x00000080); // letterbox is dark blue
```

Address: 20F\_0000h base + 100h offset = 20F\_0100h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD								COLOR																							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**PXP\_PS\_BACKGROUND field descriptions**

Field	Description
31–24 RSVD	Reserved, always set to zero.
23–0 COLOR	Background color (in 24bpp format) for any pixels not within the buffer range specified by the PS ULC/LRC.

### 38.7.18 PS Scale Factor Register (PXP\_PS\_SCALE)

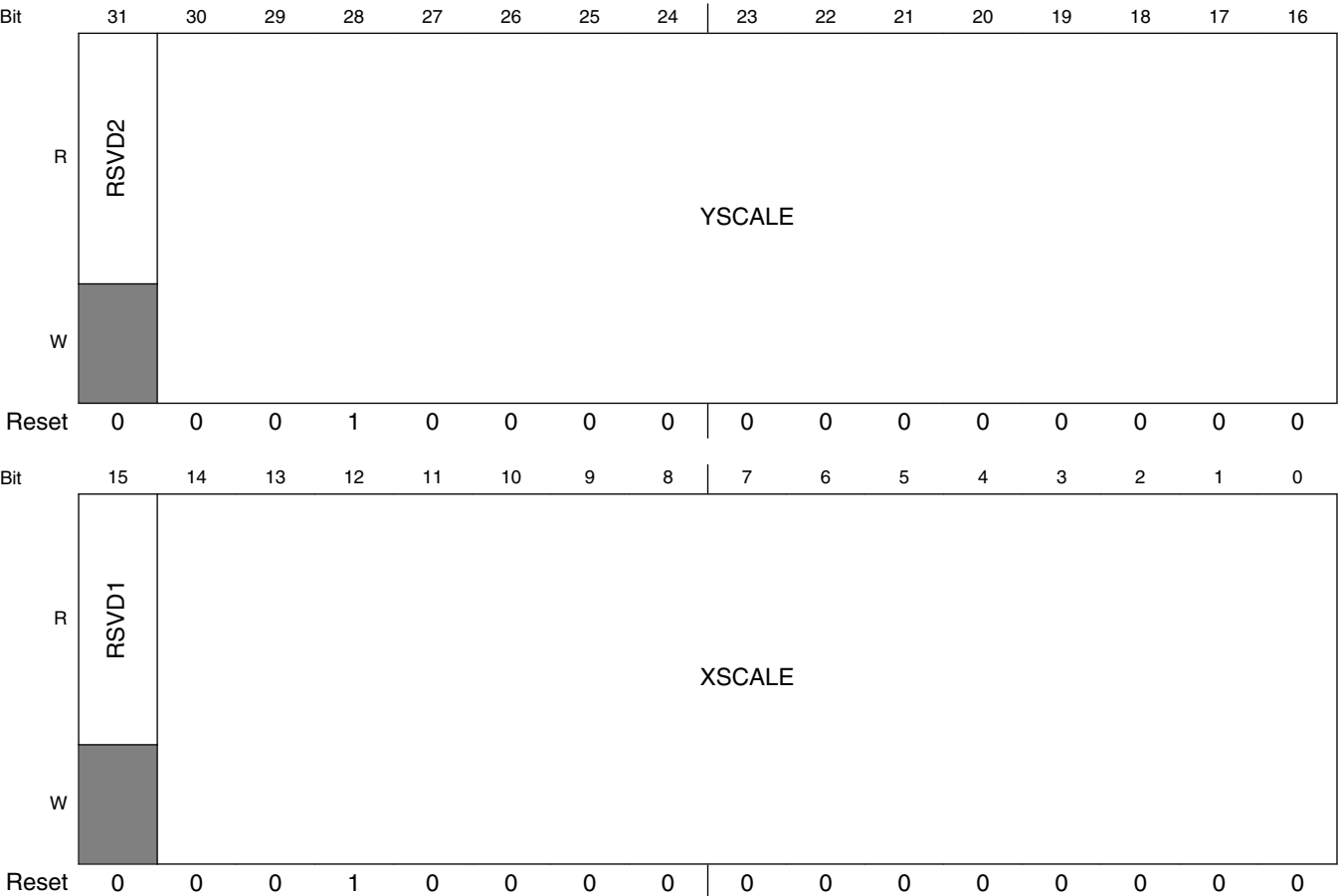
PS Scale Factor. This register provides the scale factor for the PS buffer.

The maximum down scaling factor is 1/2 such that the output image in either axis is 1/2 the size of the source. The maximum up scaling factor is  $2^{12}$  for either axis. The reciprocal of the scale factor should be loaded into this register. To reduce the PS buffer by a factor of two in the output frame buffer, a value of 10.0000\_0000\_0000 should be loaded into this register. To scale up by a factor of 4, the value of 1/4, or 00.0100\_0000\_0000, should be loaded into this register. To scale up by 8/5, the value of 00.1010\_0000\_0000 should be loaded.

EXAMPLE

```
PXP_PS_SCALE_WR(0x10001000); // 1:1 scaling (0x1.000)
PXP_PS_SCALE_WR(0x08000800); // 2x scaling (0x0.800)
PXP_PS_SCALE_WR(0x20002000); // 1/2x scaling (0x2.000)
```

Address: 20F\_0000h base + 110h offset = 20F\_0110h



PXP\_PS\_SCALE field descriptions

Field	Description
31 RSVD2	Reserved, always set to zero.
30–16 YSCALE	This is a two bit integer and 12 bit fractional representation (##.####_####_####) of the Y scaling factor for the PS source buffer. The maximum value programmed should be 2 since scaling down by a factor greater than 2 is not supported with the bilinear filter. Decimation and the bilinear filter should be used together to achieve scaling by more than a factor of 2.
15 RSVD1	Reserved, always set to zero.
14–0 XSCALE	This is a two bit integer and 12 bit fractional representation (##.####_####_####) of the X scaling factor for the PS source buffer. The maximum value programmed should be 2 since scaling down by a factor greater than 2 is not supported with the bilinear filter. Decimation and the bilinear filter should be used together to achieve scaling by more than a factor of 2.

### 38.7.19 PS Scale Offset Register (PXP\_PS\_OFFSET)

PS Scale Offset. This register provides the initial scale offset for the PS buffer.

The X and Y offset provides the ability to access the source image with a per sub-pixel granularity. This provides the capability to use all source pixels to effect the output PS image. The fixed offset values can be used for sub-pixel adjustments in the bilinear scaling filter. For example, when scaling an image down by a factor of 2, an initial offset of 0x0 would result in sub-sampling every other pixel. If a fixed offset of 0x800 (1/2), all pixels are used in scaling the final output pixel value. In this case, the first output pixel would be the sum of  $(1/2 * P_0) + (1/2 * P_1)$ . This fixed offset is applied after the decimation filter stage, and before the bilinear filter stage.

#### EXAMPLE

```
PXP_PS_SCALE_WR(0x2000_2000); // 1/2x scaling (0x2.000)
PXP_PS_OFFSET_WR(0x0800_0800); // half-pixel offset in both X and Y to ensure averaging
versus pixel decimation
```

Address: 20F\_0000h base + 120h offset = 20F\_0120h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD2				YOFFSET												RSVD1				XOFFSET											
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### PXP\_PS\_OFFSET field descriptions

Field	Description
31–28 RSVD2	Reserved, always set to zero.
27–16 YOFFSET	This is a 12 bit fractional representation (0.####_####_####) of the Y scaling offset. This represents a fixed pixel offset which gets added to the scaled address to determine source data for the scaling engine.
15–12 RSVD1	Reserved, always set to zero.
11–0 XOFFSET	This is a 12 bit fractional representation (0.####_####_####) of the X scaling offset. This represents a fixed pixel offset which gets added to the scaled address to determine source data for the scaling engine.

### 38.7.20 PS Color Key Low (PXP\_PS\_CLRKEYLOW)

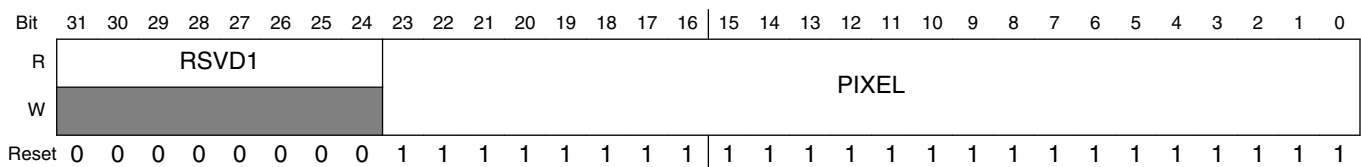
This register contains the color key low value for the PS buffer.

When processing an image, if the PXP finds a pixel in the PS buffer with a color that falls in the range between PXP\_PS\_CLRKEYLOW and PXP\_PS\_CLRKEYHIGH, it will insert the pixel from the AS channel. If the current AS pixel is letterboxed or if the AS also matches its colorkey range, the PXP\_PS\_BACKGROUND color is passed down the pixel pipeline.

### EXAMPLE

```
// colorkey values between
PXP_PS_CLRKEYLOW_WR (0x008000); // medium green and
PXP_PS_CLRKEYHIGH_WR(0x00FF00); // light green
```

Address: 20F\_0000h base + 130h offset = 20F\_0130h



**PXP\_PS\_CLRKEYLOW field descriptions**

Field	Description
31–24 RSVD1	Reserved, always set to zero.
23–0 PIXEL	Low range of color key applied to PS buffer. To disable PS colorkeying, set the low colorkey to 0xFFFFFFFF and the high colorkey to 0x000000.

### 38.7.21 PS Color Key High (PXP\_PS\_CLRKEYHIGH)

This register contains the color key high value for the PS buffer.

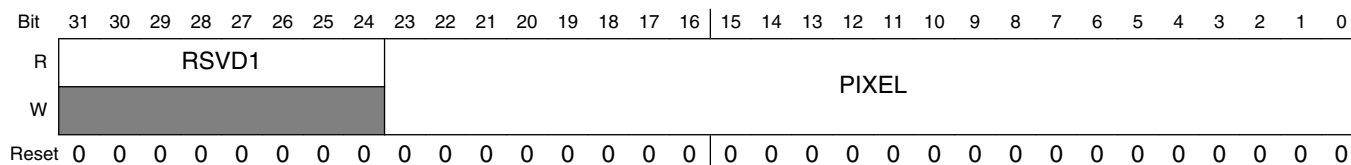
When processing an image, if the PXP finds a pixel in the PS buffer with a color that falls in the range between PXP\_PS\_CLRKEYLOW and PXP\_PS\_CLRKEYHIGH, it will insert the pixel from the AS channel. If the current AS pixel is letterboxed or if the AS also matches its colorkey range, the PXP\_PS\_BACKGROUND color is passed down the pixel pipeline.

### EXAMPLE

```
// colorkey values between
PXP_PS_CLRKEYLOW_WR (0x008000); // medium green and
PXP_PS_CLRKEYHIGH_WR(0x00FF00); // light green
```

## PXP Memory Map/Register Definition

Address: 20F\_0000h base + 140h offset = 20F\_0140h



### PXP\_PS\_CLRKEYHIGH field descriptions

Field	Description
31–24 RSVD1	Reserved, always set to zero.
23–0 PIXEL	High range of color key applied to PS buffer. To disable PS colorkeying, set the low colorkey to 0xFFFFFFFF and the high colorkey to 0x000000.

## 38.7.22 Alpha Surface Control (PXP\_AS\_CTRL)

This register contains buffer control for the Alpha Surface 0 input buffer.

The Alpha Surface Parameter register provides additional controls for AS.

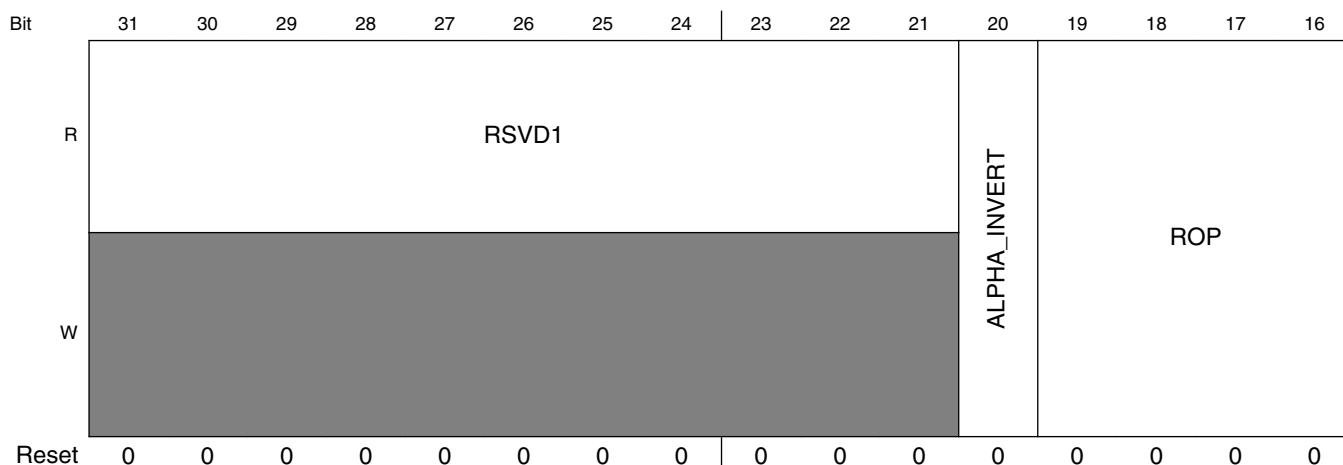
### EXAMPLE

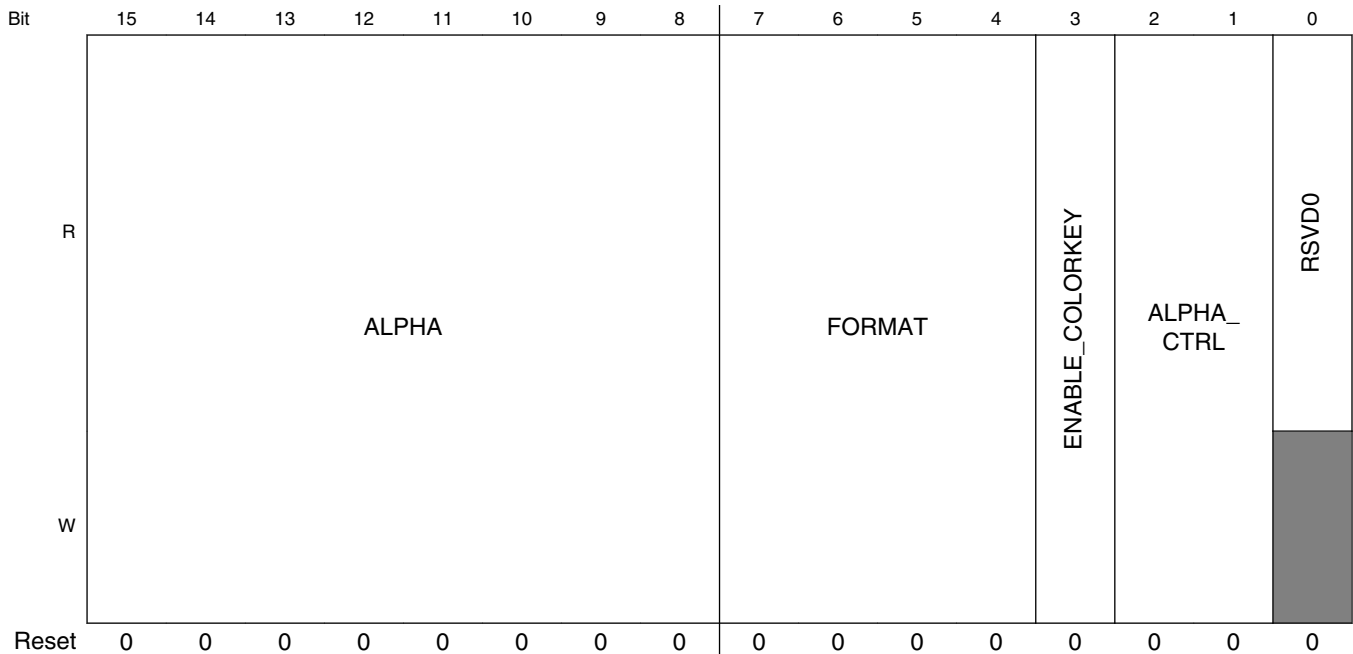
```

u32 asparam;
    asparam = BF_PXP_ASPARAM_ENABLE      (1);
    asparam |= BF_PXP_ASPARAM_ALPHA_CTRL (BV_PXP_ASPARAM_ALPHA_CTRL_ROPS);
    asparam |= BF_PXP_ASPARAM_FORMAT     (BV_PXP_ASPARAM_FORMAT_ARGB8888);
    asparam |= BF_PXP_ASPARAM_ROP       (BV_PXP_ASPARAM_ROP_XORAS);
PXP_ASPARAM_WR(0,asparam); // enable alpha surface to perform XOR ROP using RGB8888 AS
pixel format

```

Address: 20F\_0000h base + 150h offset = 20F\_0150h





PXP\_AS\_CTRL field descriptions

Field	Description
31–21 RSVD1	Reserved, always set to zero.
20 ALPHA_INVERT	Setting this bit to logic 0 will not alter the alpha value. A logic 1 will invert the alpha value and apply (1-alpha) for image composition.
19–16 ROP	Indicates a raster operation to perform when enabled. Raster operations are enabled through the ALPHA_CTRL field.  0x0 <b>MASKAS</b> — AS AND PS 0x1 <b>MASKNOTAS</b> — nAS AND PS 0x2 <b>MASKASNOT</b> — AS AND nPS 0x3 <b>MERGEAS</b> — AS OR PS 0x4 <b>MERGENOTAS</b> — nAS OR PS 0x5 <b>MERGEASNOT</b> — AS OR nPS 0x6 <b>NOTCOPYAS</b> — nAS 0x7 <b>NOT</b> — nPS 0x8 <b>NOTMASKAS</b> — AS NAND PS 0x9 <b>NOTMERGEAS</b> — AS NOR PS 0xA <b>XORAS</b> — AS XOR PS 0xB <b>NOTXORAS</b> — AS XNOR PS
15–8 ALPHA	Alpha modifier used when the ALPHA_MULTIPLY or ALPHA_OVERRIDE values are programmed in PXP_AS_CTRL[ALPHA_CTRL]. The output alpha value will either be replaced (ALPHA_OVERRIDE) or scaled (ALPHA_MULTIPLY) when selected.
7–4 FORMAT	Indicates the input buffer format for AS.  0x0 <b>ARGB8888</b> — 32-bit pixels with alpha 0x4 <b>RGB888</b> — 32-bit pixels without alpha (unpacked 24-bit format) 0x8 <b>ARGB1555</b> — 16-bit pixels with alpha

Table continues on the next page...

**PXP\_AS\_CTRL field descriptions (continued)**

Field	Description
	0x9 <b>ARGB4444</b> — 16-bit pixels with alpha 0xC <b>RGB555</b> — 16-bit pixels without alpha 0xD <b>RGB444</b> — 16-bit pixels without alpha 0xE <b>RGB565</b> — 16-bit pixels without alpha
3 ENABLE_ COLORKEY	Indicates that colorkey functionality is enabled for this alpha surface. Pixels found in the alpha surface colorkey range will be displayed as transparent (the PS pixel will be used).
2–1 ALPHA_CTRL	Determines how the alpha value is constructed for this alpha surface. Indicates that the value in the ALPHA field should be used instead of the alpha values present in the input pixels.  0x0 <b>Embedded</b> — Indicates that the AS pixel alpha value will be used to blend the AS with PS. The ALPHA field is ignored. 0x1 <b>Override</b> — Indicates that the value in the ALPHA field should be used instead of the alpha values present in the input pixels. 0x2 <b>Multiply</b> — Indicates that the value in the ALPHA field should be used to scale all pixel alpha values. Each pixel alpha is multiplied by the value in the ALPHA field. 0x3 <b>ROPs</b> — Enable ROPs. The ROP field indicates an operation to be performed on the alpha surface and PS pixels.
0 RSVD0	Reserved, always set to zero.

**38.7.23 Alpha Surface Buffer Pointer (PXP\_AS\_BUF)**

Alpha Surface 0 Buffer Address Pointer. This register points to the beginning of the Alpha Surface 0 input buffer.

This register is used to indicate the base address of the AS buffer.

**EXAMPLE**

```
u32* alpha_ptr;
PXP_ASn_WR(0, alpha_ptr);
```

Address: 20F\_0000h base + 160h offset = 20F\_0160h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**PXP\_AS\_BUF field descriptions**

Field	Description
31–0 ADDR	Address pointer for the alpha surface 0 buffer.



### 38.7.24 Alpha Surface Pitch (PXP\_AS\_PITCH)

This register contains the alpha surface pitch in bytes.

Any byte value will indicate the vertical pitch. This value will be used in AS pixel address calculations. This value has no relation to the UL and LR registers. It specifies how many bytes are between two vertically adjacent pixels in the input AS surface.

#### EXAMPLE

```
PXP_AS_PITCH_WR( 1920 * 4 ); // The output buffer pitch is HD resolution at 32 bits per pixel
```

Address: 20F\_0000h base + 170h offset = 20F\_0170h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD																PITCH															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### PXP\_AS\_PITCH field descriptions

Field	Description
31–16 RSVD	Reserved, always set to zero.
15–0 PITCH	Indicates the number of bytes in memory between two vertically adjacent pixels.

### 38.7.25 Overlay Color Key Low (PXP\_AS\_CLRKEYLOW)

This register contains the color key low value for the AS buffer.

When processing an image, the if the PXP finds a pixel in the current overlay image with a color that falls in the range from the ASCOLORKEYLOW to ASCOLORKEYHIGH range, it will use the PS pixel value for that location. If no PS image is present or if the PS image also matches its colorkey range, the PS background color is used. Colorkey operations are higher priority than alpha or ROP operations.

#### EXAMPLE

```
// colorkey values between
PXP_AS_CLRKEYLOW_WR (0x000000); // black and
PXP_AS_CLRKEYHIGH_WR(0x800000); // medium red
```

## PXP Memory Map/Register Definition

Address: 20F\_0000h base + 180h offset = 20F\_0180h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD1								PIXEL																							
W																																
Reset	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	

### PXP\_AS\_CLRKEYLOW field descriptions

Field	Description
31–24 RSVD1	Reserved, always set to zero.
23–0 PIXEL	Low range of RGB color key applied to AS buffer. Each overlay has an independent colorkey enable.

## 38.7.26 Overlay Color Key High (PXP\_AS\_CLRKEYHIGH)

This register contains the color key high value for the AS buffer.

When processing an image, the if the PXP finds a pixel in the current overlay image with a color that falls in the range from the ASCOLORKEYLOW to ASCOLORKEYHIGH range, it will use the PS pixel value for that location. If no PS image is present or if the PS image also matches its colorkey range, the PS background color is used. Colorkey operations are higher priority than alpha or ROP operations.

### EXAMPLE

```
// colorkey values between  
PXP_AS_CLRKEYLOW_WR (0x000000); // black and  
PXP_AS_CLRKEYHIGH_WR(0x800000); // medium red
```

Address: 20F\_0000h base + 190h offset = 20F\_0190h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD1								PIXEL																							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### PXP\_AS\_CLRKEYHIGH field descriptions

Field	Description
31–24 RSVD1	Reserved, always set to zero.
23–0 PIXEL	High range of RGB color key applied to AS buffer. Each overlay has an independent colorkey enable.

### 38.7.27 Color Space Conversion Coefficient Register 0 (PXP\_CSC1\_COEF0)

This register contains color space conversion coefficients in two's complement notation.

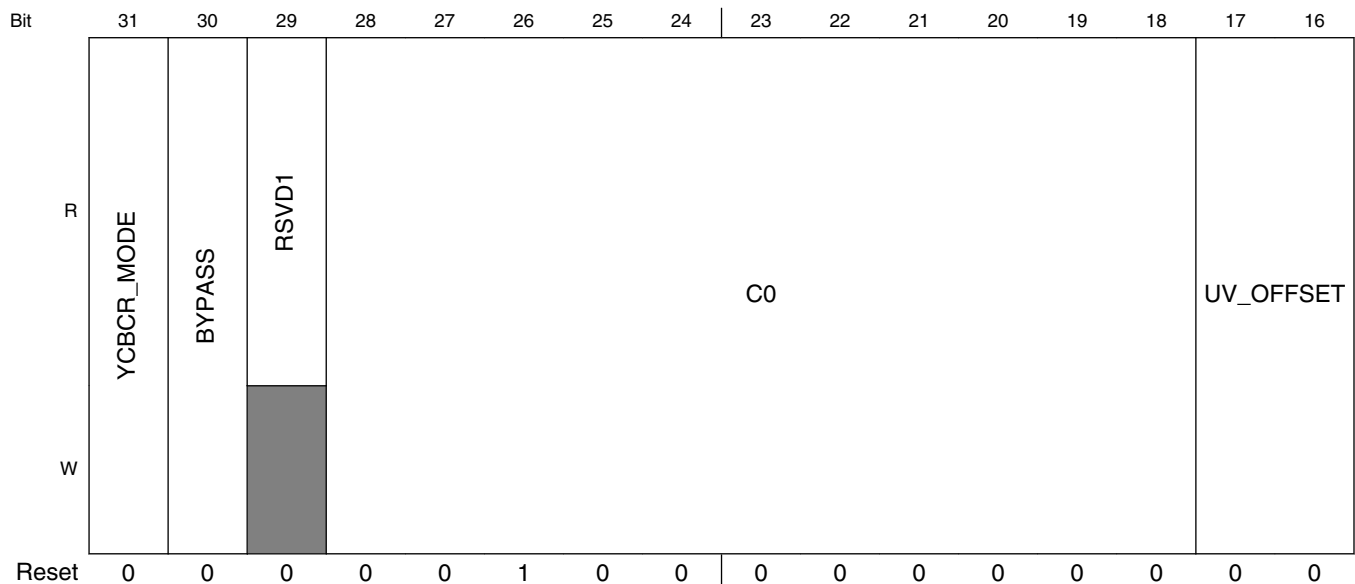
The coefficient 0 register contains coefficients used in the color space conversion algorithm. The Y and UV offsets are added to the source buffer to normalize them before the conversion. C0 is the coefficient that is used to multiply the luma component of the data for all three RGB components.

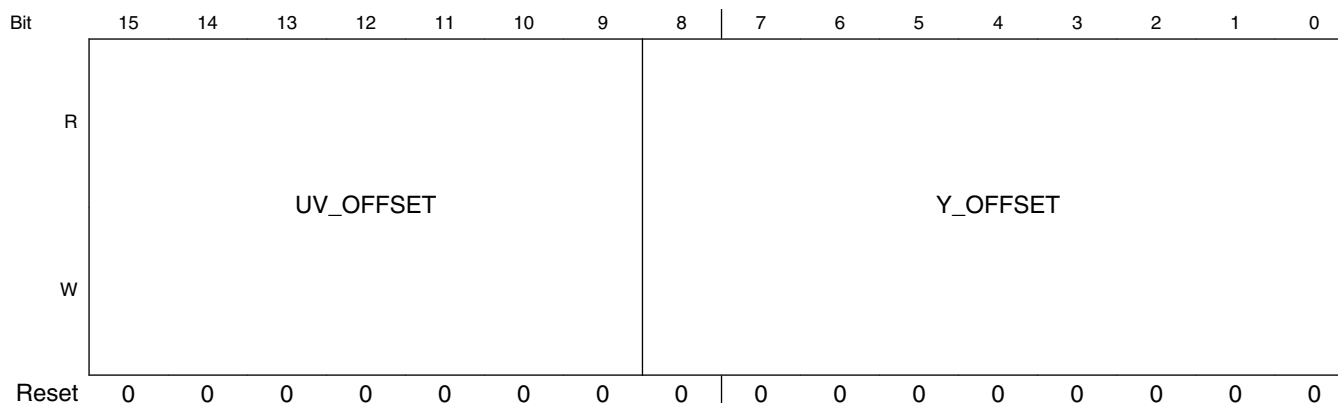
#### EXAMPLE

```
// The equations used for Colorspace conversion are:
//   R = C0*(Y+YOFFSET) + C1(V+UV_OFFSET)
//   G = C0*(Y+YOFFSET) + C3(U+UV_OFFSET) + C2(V+UV_OFFSET)
//   B = C0*(Y+YOFFSET) + C4(U+UV_OFFSET)

PXP_CSCCOEF0_WR(0x04030000); // YUV coefficients: C0, Yoffset, UVoffset
PXP_CSCCOEF1_WR(0x01230208); // YUV coefficients: C1, C4
PXP_CSCCOEF2_WR(0x076B079b); // YUV coefficients: C2, C3
```

Address: 20F\_0000h base + 1A0h offset = 20F\_01A0h





**PXP\_CSC1\_COEF0 field descriptions**

Field	Description
31 YCBCR_MODE	Set to 1 when performing YCbCr conversion to RGB. Set to 0 when converting YUV to RGB data. This bit changes the behavior of the scaler when performing U/V scaling.
30 BYPASS	Bypass the CSC unit in the scaling engine. When set to logic 1, bypass is enabled and the output pixels will be in the YUV/YCbCr color space. When set to logic 0, the CSC unit is enabled and the pixels will be converted based on the programmed coefficients.
29 RSVD1	Reserved, always set to zero.
28–18 C0	Two's compliment Y multiplier coefficient. YUV=0x100 (1.000) YCbCr=0x12A (1.164)
17–9 UV_OFFSET	Two's compliment phase offset implicit for CbCr data. Generally used for YCbCr to RGB conversion. YCbCr=0x180, YUV=0x000 (typically -128 or 0x180 to indicate normalized -0.5 to 0.5 range)
8–0 Y_OFFSET	Two's compliment amplitude offset implicit in the Y data. For YUV, this is typically 0 and for YCbCr, this is typically -16 (0x1F0)

## 38.7.28 Color Space Conversion Coefficient Register 1 (PXP\_CSC1\_COEF1)

This register contains color space conversion coefficients in two's compliment notation.

The Coefficient 1 register contains coefficients used in the color space conversion algorithm. C1 is the coefficient that is used to multiply the chroma (Cr/V) component of the data for the red component. C4 is the coefficient that is used to multiply the chroma (Cb/U) component of the data for the blue component. Both values should be coded as a two's compliment fixed point number with 8 bits right of the decimal.

### EXAMPLE

```
PXP_CSCCOEF0_WR(0x04030000); // YUV coefficients: C0, Yoffset, UYoffset
PXP_CSCCOEF1_WR(0x01230208); // YUV coefficients: C1, C4
PXP_CSCCOEF2_WR(0x076B079b); // YUV coefficients: C2, C3
```

Address: 20F\_0000h base + 1B0h offset = 20F\_01B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD1					C1											RSVD0					C4										
W																																
Reset	0	0	0	0	0	0	0	1	0	0	1	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0

**PXP\_CSC1\_COEF1 field descriptions**

Field	Description
31–27 RSVD1	Reserved, always set to zero.
26–16 C1	Two's compliment Red V/Cr multiplier coefficient. YUV=0x123 (1.140) YCbCr=0x198 (1.596)
15–11 RSVD0	Reserved, always set to zero.
10–0 C4	Two's compliment Blue U/Cb multiplier coefficient. YUV=0x208 (2.032) YCbCr=0x204 (2.017)

### 38.7.29 Color Space Conversion Coefficient Register 2 (PXP\_CSC1\_COEF2)

This register contains color space conversion coefficients in two's compliment notation.

The Coefficient 2 register contains coefficients used in the color space conversion algorithm. C2 is the coefficient that is used to multiply the chroma (Cr/V) component of the data for the green component. C3 is the coefficient that is used to multiply the chroma (Cb/U) component of the data for the green component. Both values should be coded as a two's compliment fixed point number with 8 bits right of the decimal.

**EXAMPLE**

// NOTE: The default values for the CSCCOEF2 register are incorrect. C2 should be 0x76B and C3 should be 0x79C for proper operation.

```
PXP_CSCCOEF0_WR(0x04030000); // YUV coefficients: C0, Yoffset, UYoffset
PXP_CSCCOEF1_WR(0x01230208); // YUV coefficients: C1, C4
PXP_CSCCOEF2_WR(0x076B079b); // YUV coefficients: C2, C3
```

Address: 20F\_0000h base + 1C0h offset = 20F\_01C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD1					C2										RSVD0					C3											
W																																
Reset	0	0	0	0	0	1	1	1	1	0	0	1	1	0	1	1	0	0	0	0	0	1	1	1	0	1	1	0	1	1	0	0

## PXP\_CSC1\_COEF2 field descriptions

Field	Description
31–27 RSVD1	Reserved, always set to zero.
26–16 C2	Two's complement Green V/Cr multiplier coefficient. YUV=0x76B (-0.581) YCbCr=0x730 (-0.813)
15–11 RSVD0	Reserved, always set to zero.
10–0 C3	Two's complement Green U/Cb multiplier coefficient. YUV=0x79C (-0.394) YCbCr=0x79C (-0.392)

### 38.7.30 Color Space Conversion Control Register. (PXP\_CSC2\_CTRL)

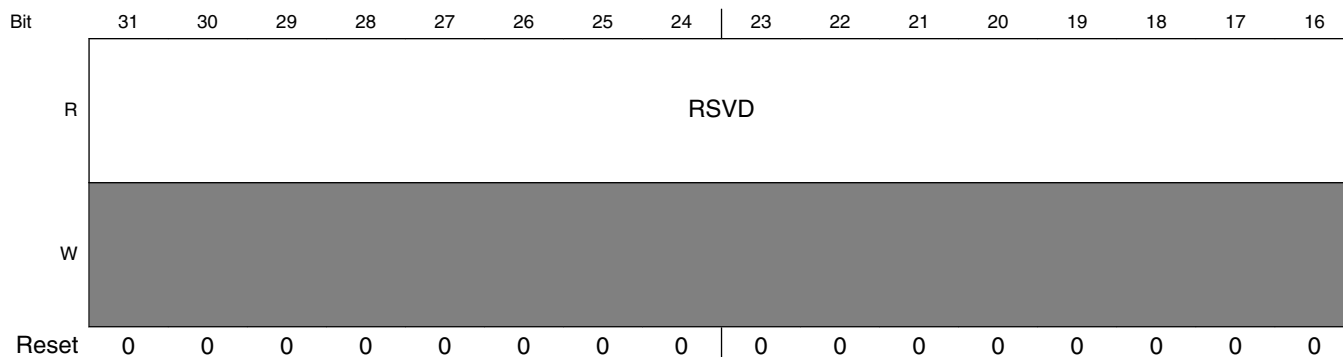
This register contains the control registers to configure the CSC module.

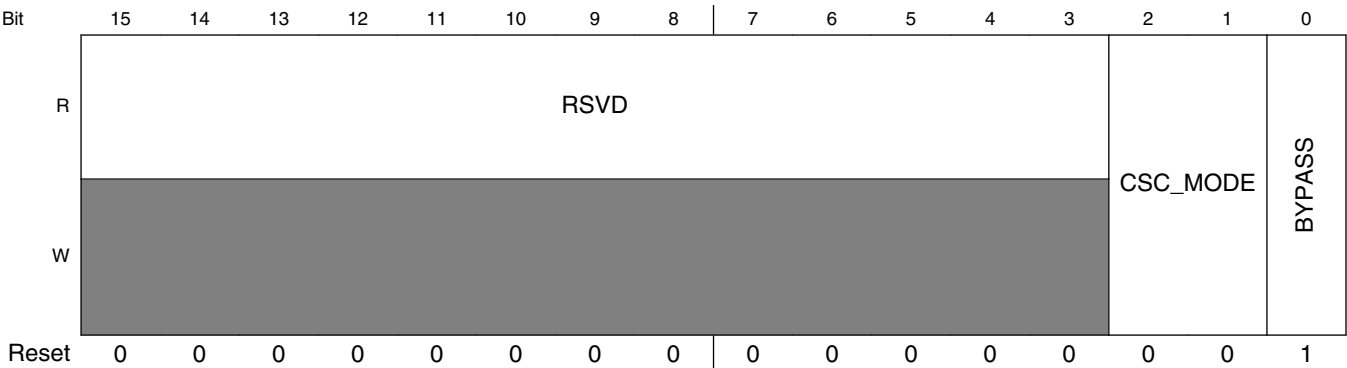
The CSC control register will configure the CSC module to perform color space conversion between the RGB/YUV/YCbCr color spaces.

#### EXAMPLE

```
//Converting from YUV/YCbCr color spaces to the RGB color space uses the
//following equation structure:
//
// R = A1(Y-D1) + A2(U-D2) + A3(V-D3)
// G = B1(Y-D1) + B2(U-D2) + B3(V-D3)
// B = C1(Y-D1) + C2(U-D2) + C3(V-D3)
//
//Converting from the RGB color space to YUV/YCbCr color spaces uses the
//following equation structure:
//
// Y = A1*R + A2*G + A3*B + D1
// U = B1*R + B2*G + B3*B + D2
// V = C1*R + C2*G + C3*B + D3
//
//All math is signed, so all coefficients come in as two's comp numbers
//
```

Address: 20F\_0000h base + 1D0h offset = 20F\_01D0h





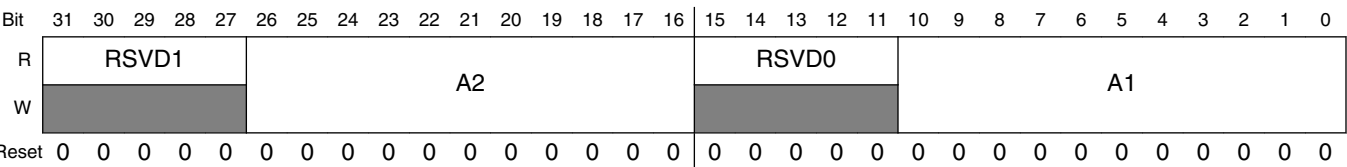
PXP\_CSC2\_CTRL field descriptions

Field	Description
31–3 RSVD	Reserved, always set to zero.
2–1 CSC_MODE	This field controls how the CSC unit operates on pixels when the CSC is not bypassed. 0x0 <b>YUV2RGB</b> — Convert from YUV to RGB. 0x1 <b>YCbCr2RGB</b> — Convert from YCbCr to RGB. 0x2 <b>RGB2YUV</b> — Convert from RGB to YUV. 0x3 <b>RGB2YCbCr</b> — Convert from RGB to YCbCr.
0 BYPASS	This bit controls whether the pixels entering the CSC2 unit get converted or not. When BYPASS is set, no operations occur on the pixels. When BYPASS is cleared, the selected CSC operation takes place.

38.7.31 Color Space Conversion Coefficient Register 0 (PXP\_CSC2\_COEF0)

This register contains color space conversion coefficients in two's complement notation.

Address: 20F\_0000h base + 1E0h offset = 20F\_01E0h



**PXP\_CSC2\_COEF0 field descriptions**

Field	Description
31–27 RSVD1	Reserved, always set to zero.
26–16 A2	Two's complement coefficient offset. This coefficient has a sign bit, 2 bits integer, and 8 bits of fraction as ###.####_####.
15–11 RSVD0	Reserved, always set to zero.
10–0 A1	Two's complement coefficient offset. This coefficient has a sign bit, 2 bits integer, and 8 bits of fraction as ###.####_####.

### 38.7.32 Color Space Conversion Coefficient Register 1 (PXP\_CSC2\_COEF1)

This register contains color space conversion coefficients in two's complement notation.

Address: 20F\_0000h base + 1F0h offset = 20F\_01F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD1					B1											RSVD0					A3										
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**PXP\_CSC2\_COEF1 field descriptions**

Field	Description
31–27 RSVD1	Reserved, always set to zero.
26–16 B1	Two's complement coefficient offset. This coefficient has a sign bit, 2 bits integer, and 8 bits of fraction as ###.####_####.
15–11 RSVD0	Reserved, always set to zero.
10–0 A3	Two's complement coefficient offset. This coefficient has a sign bit, 2 bits integer, and 8 bits of fraction as ###.####_####.

### 38.7.33 Color Space Conversion Coefficient Register 2 (PXP\_CSC2\_COEF2)

This register contains color space conversion coefficients in two's complement notation.



Address: 20F\_0000h base + 200h offset = 20F\_0200h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD1					B3											RSVD0					B2										
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### PXP\_CSC2\_COEF2 field descriptions

Field	Description
31–27 RSVD1	Reserved, always set to zero.
26–16 B3	Two's complement coefficient offset. This coefficient has a sign bit, 2 bits integer, and 8 bits of fraction as ###.####_####.
15–11 RSVD0	Reserved, always set to zero.
10–0 B2	Two's complement coefficient offset. This coefficient has a sign bit, 2 bits integer, and 8 bits of fraction as ###.####_####.

## 38.7.34 Color Space Conversion Coefficient Register 3 (PXP\_CSC2\_COEF3)

This register contains color space conversion coefficients in two's complement notation.

Address: 20F\_0000h base + 210h offset = 20F\_0210h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD1					C2											RSVD0					C1										
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### PXP\_CSC2\_COEF3 field descriptions

Field	Description
31–27 RSVD1	Reserved, always set to zero.
26–16 C2	Two's complement coefficient offset. This coefficient has a sign bit, 2 bits integer, and 8 bits of fraction as ###.####_####.

*Table continues on the next page...*

**PXP\_CSC2\_COEF3 field descriptions (continued)**

Field	Description
15–11 RSVD0	Reserved, always set to zero.
10–0 C1	Two's complement coefficient offset. This coefficient has a sign bit, 2 bits integer, and 8 bits of fraction as ###.####_####.

### 38.7.35 Color Space Conversion Coefficient Register 4 (PXP\_CSC2\_COEF4)

This register contains color space conversion coefficients in two's complement notation.

Address: 20F\_0000h base + 220h offset = 20F\_0220h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD1								D1								RSVD0				C3											
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**PXP\_CSC2\_COEF4 field descriptions**

Field	Description
31–25 RSVD1	Reserved, always set to zero.
24–16 D1	Two's complement coefficient integer offset to be added.
15–11 RSVD0	Reserved, always set to zero.
10–0 C3	Two's complement coefficient offset. This coefficient has a sign bit, 2 bits integer, and 8 bits of fraction as ###.####_####.

### 38.7.36 Color Space Conversion Coefficient Register 5 (PXP\_CSC2\_COEF5)

This register contains color space conversion coefficients in two's complement notation.

Address: 20F\_0000h base + 230h offset = 20F\_0230h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD1								D3								RSVD0								D2							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### PXP\_CSC2\_COEF5 field descriptions

Field	Description
31–25 RSVD1	Reserved, always set to zero.
24–16 D3	Two's complement coefficient integer offset to be added.
15–9 RSVD0	Reserved, always set to zero.
8–0 D2	Two's complement D1 coefficient integer offset to be added.

## 38.7.37 Lookup Table Control Register. (PXP\_LUT\_CTRL)

This register is used to access/control the Monochrome Lookup table.

The Y8 LUT input mode will take the high order data path byte and transform it using the LUT memory. This is an 8-bit to 8-bit transformation. The two low order bytes bypass the LUT and are not transformed, but bypassed without modification. This option can be used for monochrome gamma correction. The Direct Lookup mode will use the high nibble of each data byte and truncate the low nibble to generate the lookup address, i.e. R[7:0]G[7:0]B[7:0] -> R[7:4]G[7:4]B[7:4]. 4K pixels (12-bit address) with 2 bytes per pixel is supported in this mode. Cached Lookup mode will use the high order bits, R[7:3],G[7:2],B[7:3] or RGB565, to address the cached LUT memory. 64KB LUT tables, using 16KB of internal LUT memory, can be indirectly transformed to 16-bit output pixels (as in RGBW4444/RGB565). This is used for 16bpp gamma correction or EPD color panel support. Cache misses are internally managed by the PXP LUT Cache controller.

## PXP Memory Map/Register Definition

Address: 20F\_0000h base + 240h offset = 20F\_0240h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BYPASS	RSVD3					LOOKUP_MODE		RSVD2							OUT_MODE
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD1					SEL_8KB	LRU_UPD	INVALID	RSVD0							DMA_START
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### PXP\_LUT\_CTRL field descriptions

Field	Description
31 BYPASS	Setting this bit will bypass the LUT memory resource completely. No pixel transformations will occur at this stage of the PXP pixel rprocessing pipeline.
30–26 RSVD3	Reserved, always set to zero.
25–24 LOOKUP_MODE	Configure the input address for the 16KB LUT memory. The address into the LUT uses different parts of the pixel data path bytes. The data path is defined as three bytes, conceptually as RGB/YUV/YCbCr[23:0]. Also referred to as R/Y[7:0],G/U[7:0],B/V[7:0]  0x0 <b>CACHE_RGB565</b> — LUT ADDR = R[7:3],G[7:2],B[7:3]. Use all 16KB of LUT for indirect cached 128KB lookup. 0x1 <b>DIRECT_Y8</b> — LUT ADDR = 16'b0,Y[7:0]. Use only the first 256 bytes of LUT. Only the Y, or third data path byte, is tranformed. 0x2 <b>DIRECT_RGB444</b> — LUT ADDR = R[7:4],G[7:4],B[7:4]. Use one 8KB bank of LUT selected by SEL_8KB. 0x3 <b>DIRECT_RGB454</b> — LUT ADDR = R[7:4],G[7:3],B[7:4]. Use all 16KB of LUT.
23–18 RSVD2	Reserved, always set to zero.

Table continues on the next page...

**PXP\_LUT\_CTRL field descriptions (continued)**

Field	Description
17–16 OUT_MODE	Select the output mode of operation for the LUT resource. There are four bytes [3-0] in the data path at the output of the LUT resource. Byte lane 3 is always bypassed and usually contains an alpha value. The LUT can be programmed to transform bytes 2,1,0 according to the options available in this field.  0x0 <b>RESERVED</b> — Reserved, not valid when using the LUT to transform pixels. 0x1 <b>Y8</b> — R/Y byte lane 2 lookup, bytes 1,0 bypassed. 0x2 <b>RGBW4444CFA</b> — Byte lane 2 = CFA_Y8, byte lane 1,0 = RGBW4444. 0x3 <b>RGB888</b> — RGB565->RGB888 conversion for Gamma correction.
15–11 RSVD1	Reserved, always set to zero.
10 SEL_8KB	Selects which 8KB bank of memory to use for direct 12bpp lookup modes. Logic 0 indicates first 8KB, logic 1 indicates second 8KB. Two direct LUT arrays can be stored and one can be selected for a given PXP operation.
9 LRU_UPD	Least Recently Used Policy Update Control: 1=> block LRU update for hit after miss. 0=> update LRU for all hits including hit after miss.
8 INVALID	Invalidate the cache LRU and valid bits. This bit will automatically reset when set to a logic 1.
7–1 RSVD0	Reserved, always set to zero.
0 DMA_START	Setting this bit will result in the DMA operation to load the PXP LUT memory based on PXP_LUT_ADDR_NUM_BYTES, PXP_LUT_ADDR_ADDR, and PXP_LUT_MEM_ADDR.  This bit will automatically reset when set to a logic 1. Note: The LOOKUP_MODE must not be set to CACHE_RGB565 when starting and performing DMA transfers.

**38.7.38 Lookup Table Control Register. (PXP\_LUT\_ADDR)**

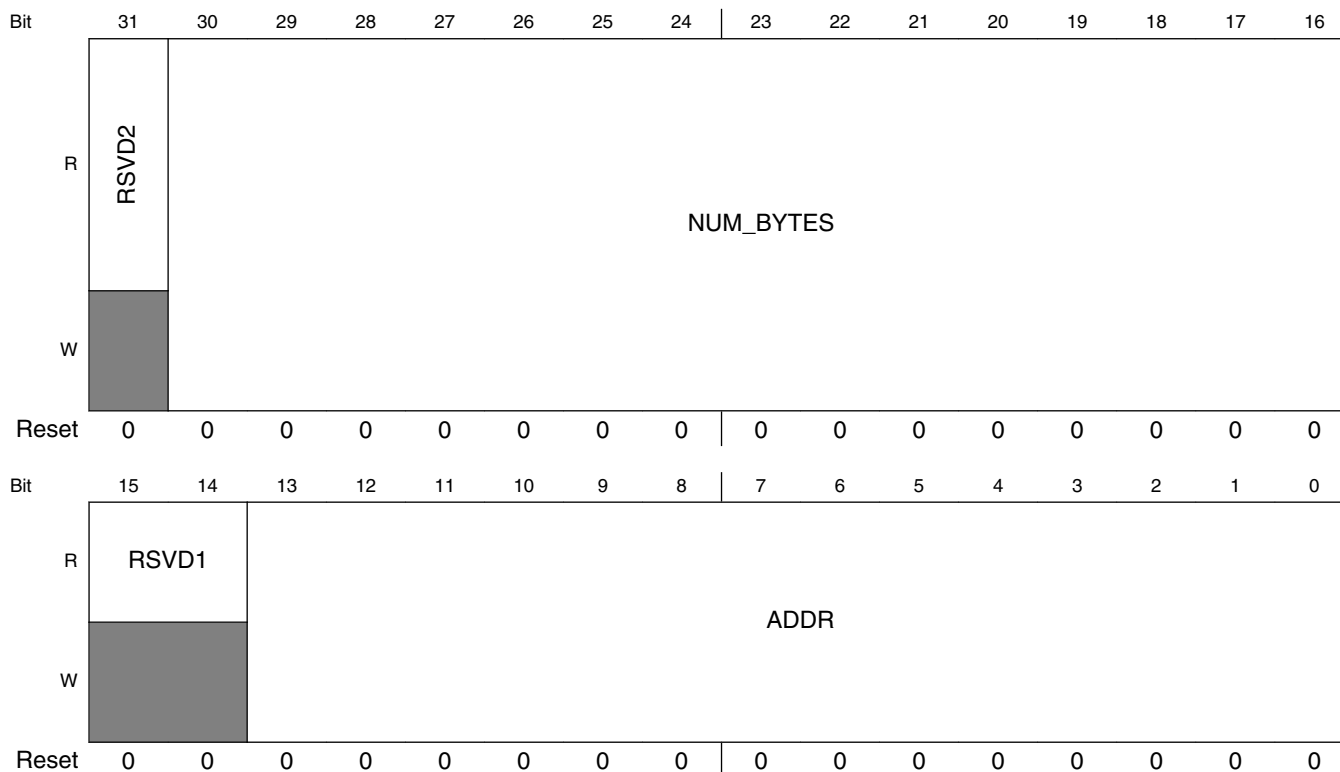
This register is used to access/control the Monochrome Lookup table.

The Y8 LUT input mode will take the high order data path byte and transform it using the LUT memory. This is an 8-bit to 8-bit transformation. The two low order bytes bypass the LUT and are not transformed, but bypassed without modification. This option can be used for monochrome gamma correction. The Direct Lookup mode will use the high nibble of each data byte and truncate the low nibble to generate the lookup address, i.e. R[7:0]G[7:0]B[7:0] -> R[7:4]G[7:4]B[7:4]. 4K pixels (12-bit address) with 2 bytes per pixel is supported in this mode. Cached Lookup mode will use the high order bits, R[7:3],G[7:2],B[7:3] or RGB565, to address the cached LUT memory. 64KB LUT tables, using 16KB of internal LUT memory, can be indirectly transformed to 16-bit output

## PXP Memory Map/Register Definition

pixels (as in RGBW4444/RGB565). This is used for 16bpp gamma correction or EPD color panel support. Cache misses are internally managed by the PXP LUT Cache controller.

Address: 20F\_0000h base + 250h offset = 20F\_0250h



### PXP\_LUT\_ADDR field descriptions

Field	Description
31 RSVD2	Reserved, always set to zero.
30–16 NUM_BYTES	Indicates the number of bytes to load via a DMA operation. This field must be divisible by 8 and the least significant 3 bits must be 0. The value 8 indicates load 8 bytes from the external address indicated by PXP_LUT_MEM_ADDR to the LUT memory location indicated by PXP_LUT_CTRL_ADDR.
15–14 RSVD1	Reserved, always set to zero.
13–0 ADDR	LUT indexed address pointer. This address into the LUT memory is always four byte aligned for PIO access, and eight byte aligned for DMA access.  The least two significant bits are not used to drive the LUT memory array. For PIO LUT access, when the LUT data register is written, the contents of the LUT at the address specified by this address field will be loaded with a 32-bit DWORD. This address pointer will be incremented after the LUT data is written. This will provide recursive writes to the LUT data register to initialize the entire LUT array with recursive writes to the LUT data register. For DMA access, this register indicates the LUT memory address of the 8 byte QWORD to be loaded. When using the NUM_BYTES field to load more than 8 bytes, the register should be programmed with the first LUT memory location to be filled and each load of the LUT memory will increment this address field until NUM_BYTES has been loaded.

### 38.7.39 Lookup Table Data Register. (PXP\_LUT\_DATA)

This register is used to load data into the lookup table.

Address: 20F\_0000h base + 260h offset = 20F\_0260h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

#### PXP\_LUT\_DATA field descriptions

Field	Description
31–0 DATA	Writing this field will load 4 bytes, aligned to four byte boundaries, of data indexed by the ADDR field of the PXP_LUT_CTRL register.

### 38.7.40 Lookup Table External Memory Address Register. (PXP\_LUT\_EXTMEM)

For DMA LUT memory loads, this is the base address from which data will be sourced to store into the LUT memory array. For Cached LUT memory pixel transformations, this register will store the base address of the full 64K pixel LUT translation table.

Address: 20F\_0000h base + 270h offset = 20F\_0270h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

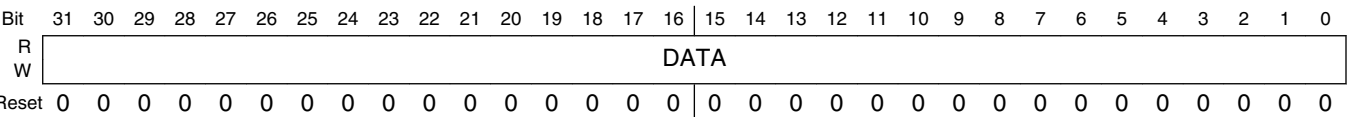
#### PXP\_LUT\_EXTMEM field descriptions

Field	Description
31–0 ADDR	This register contains the external memory address used for LUT memory operation. For DMA LUT memory loads, this is the base address from which data will be sourced to store into the LUT memory array. For Cached LUT memory pixel transformations, this register will store the base address of the full 64K pixel LUT translation table.

38.7.41 Color Filter Array Register. (PXP\_CFA)

There are sixteen 2 bit values in this register each mapping a selected component to the output pixel. The two bit values are defined as 0=>R, 1=>G, 2=>B, and 3=>W. The first byte represents the repetitive pattern of RGBW pixels in the CFA for the first line segment of each processed PXP block. The second byte represents the pattern in the second line segment of the block, and so on. The first byte repeats two times for 8x8 macro block mode, and repeats four times for 16x16 block mode.

Address: 20F\_0000h base + 280h offset = 20F\_0280h



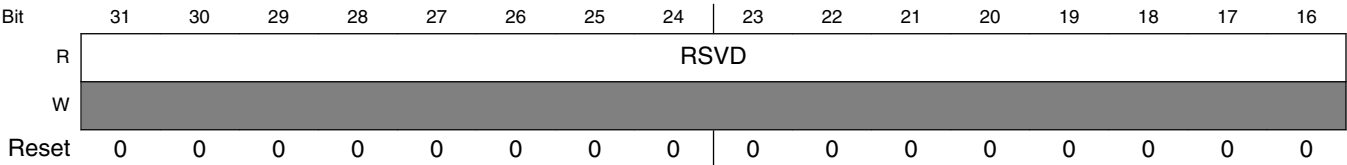
PXP\_CFA field descriptions

Field	Description
31–0 DATA	This register contains the Color Filter Array pattern for decimation of RGBW4444 16 bit pixels to individual R, G, B, W values. The pattern represents a replicated 4x4 color filter array for the entire output frame buffer.

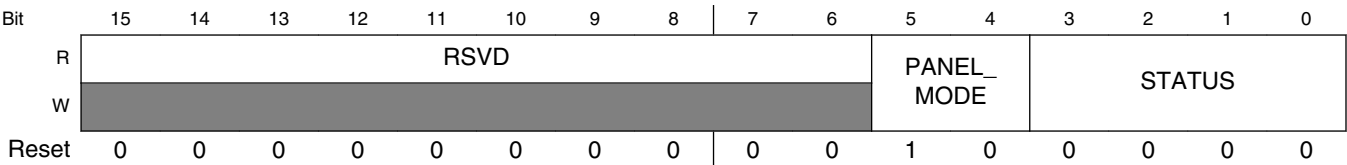
38.7.42 Histogram Control Register. (PXP\_HIST\_CTRL)

Provides control and status registers for the PXP's histogram classification algorithm.

Address: 20F\_0000h base + 290h offset = 20F\_0290h







PXP\_HIST\_CTRL field descriptions

Field	Description
31–6 RSVD	Reserved, always set to zero.
5–4 PANEL_MODE	<p>Specifies the EPDC panel grayscale depth. This value is used to specify the number of bits used in comparisons when matching pixels to histogram bins.</p> <p>All comparator values <b>MUST</b> be programmed such that their bit width is consistent with the value of this register field.</p> <p>For instance, if GRAY16 is selected, comparator values must be in the range of 0x0-0xF.</p> <p>0x0   <b>GRAY4</b> — 4-bit grayscale 0x1   <b>GRAY8</b> — 8-bit grayscale 0x2   <b>GRAY16</b> — 16-bit grayscale 0x3   <b>GRAY32</b> — 32-bit grayscale</p>
3–0 STATUS	<p>Indicates which histogram matched the processed bitmap.</p> <p>Bit[0] indicates that the bitmap pixels were fully contained within the HIST2 (black / white) histogram.</p> <p>Bit[1] indicates that the bitmap pixels were fully contained within the HIST4 (2-bit grayscale) histogram.</p> <p>Bit[2] indicates that the bitmap pixels were fully contained within the HIST8 (3-bit grayscale) histogram.</p> <p>Bit[3] indicates that the bitmap pixels were fully contained within the HIST16 (4-bit grayscale) histogram.</p>

38.7.43 2-level Histogram Parameter Register.  
(PXP\_HIST2\_PARAM)

This register specifies the valid values for a 2-level histogram. If all pixels in a bitmap match the 2-level histogram values, STATUS[0] will be set at the end of frame processing. All comparator values should be programmed such that they are consistent with the PANEL\_MODE control field.

## PXP Memory Map/Register Definition

Address: 20F\_0000h base + 2A0h offset = 20F\_02A0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD																RSVD1				VALUE1				RSVD0				VALUE0			
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

### PXP\_HIST2\_PARAM field descriptions

Field	Description
31–16 RSVD	Reserved, always set to zero.
15–13 RSVD1	Reserved, always set to zero.
12–8 VALUE1	White value for 2-level histogram
7–5 RSVD0	Reserved, always set to zero.
4–0 VALUE0	Black value for 2-level histogram

## 38.7.44 4-level Histogram Parameter Register. (PXP\_HIST4\_PARAM)

This register specifies the valid values for a 4-level histogram. If all pixels in a bitmap match the 4-level histogram values, STATUS[1] will be set at the end of frame processing. All comparator values should be programmed such that they are consistent with the PANEL\_MODE control field.

Address: 20F\_0000h base + 2B0h offset = 20F\_02B0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
R	RSVD3				VALUE3								RSVD2				VALUE2				RSVD1				VALUE1				RSVD0				VALUE0			
W																																				
Reset	0	0	0	0	1	1	1	1	0	0	0	0	1	0	1	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0	0				

### PXP\_HIST4\_PARAM field descriptions

Field	Description
31–29 RSVD3	Reserved, always set to zero.

Table continues on the next page...

**PXP\_HIST4\_PARAM field descriptions (continued)**

Field	Description
28–24 VALUE3	GRAY3 (White) value for 4-level histogram
23–21 RSVD2	Reserved, always set to zero.
20–16 VALUE2	GRAY2 value for 4-level histogram
15–13 RSVD1	Reserved, always set to zero.
12–8 VALUE1	GRAY1 value for 4-level histogram
7–5 RSVD0	Reserved, always set to zero.
4–0 VALUE0	GRAY0 (Black) value for 4-level histogram

**38.7.45 8-level Histogram Parameter 0 Register.  
(PXP\_HIST8\_PARAM0)**

This register specifies four of the valid values for an 8-level histogram. If all pixels in a bitmap match the 8-level histogram values, STATUS[2] will be set at the end of frame processing. All comparator values should be programmed such that they are consistent with the PANEL\_MODE control field.

Address: 20F\_0000h base + 2C0h offset = 20F\_02C0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD3			VALUE3				RSVD2			VALUE2				RSVD1			VALUE1				RSVD0			VALUE0							
W																																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	

**PXP\_HIST8\_PARAM0 field descriptions**

Field	Description
31–29 RSVD3	Reserved, always set to zero.
28–24 VALUE3	GRAY3 value for 8-level histogram

*Table continues on the next page...*

**PXP\_HIST8\_PARAM0 field descriptions (continued)**

Field	Description
23–21 RSVD2	Reserved, always set to zero.
20–16 VALUE2	GRAY2 value for 8-level histogram
15–13 RSVD1	Reserved, always set to zero.
12–8 VALUE1	GRAY1 value for 8-level histogram
7–5 RSVD0	Reserved, always set to zero.
4–0 VALUE0	GRAY0 (Black) value for 8-level histogram

### 38.7.46 8-level Histogram Parameter 1 Register. (PXP\_HIST8\_PARAM1)

This register specifies four of the valid values for an 8-level histogram. If all pixels in a bitmap match the 8-level histogram values, STATUS[2] will be set at the end of frame processing. All comparator values should be programmed such that they are consistent with the PANEL\_MODE control field.

Address: 20F\_0000h base + 2D0h offset = 20F\_02D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD7			VALUE7								RSVD6			VALUE6				RSVD5			VALUE5				RSVD4			VALUE4			
W																																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	1	1	0	1	0	0	0	0	1	0	1	1	0	0	0	0	1	0	0	1

**PXP\_HIST8\_PARAM1 field descriptions**

Field	Description
31–29 RSVD7	Reserved, always set to zero.
28–24 VALUE7	GRAY7 (White) value for 8-level histogram
23–21 RSVD6	Reserved, always set to zero.

*Table continues on the next page...*

**PXP\_HIST8\_PARAM1 field descriptions (continued)**

Field	Description
20–16 VALUE6	GRAY6 value for 8-level histogram
15–13 RSVD5	Reserved, always set to zero.
12–8 VALUE5	GRAY5 value for 8-level histogram
7–5 RSVD4	Reserved, always set to zero.
4–0 VALUE4	GRAY4 value for 8-level histogram

**38.7.47 16-level Histogram Parameter 0 Register.  
(PXP\_HIST16\_PARAM0)**

This register specifies four of the valid values for a 16-level histogram. If all pixels in a bitmap match the 16-level histogram values, STATUS[3] will be set at the end of frame processing. All comparator values should be programmed such that they are consistent with the PANEL\_MODE control field.

Address: 20F\_0000h base + 2E0h offset = 20F\_02E0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD3			VALUE3					RSVD2			VALUE2				RSVD1				VALUE1				RSVD0			VALUE0					
W																																
Reset	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

**PXP\_HIST16\_PARAM0 field descriptions**

Field	Description
31–29 RSVD3	Reserved, always set to zero.
28–24 VALUE3	GRAY3 value for 16-level histogram
23–21 RSVD2	Reserved, always set to zero.
20–16 VALUE2	GRAY2 value for 16-level histogram

*Table continues on the next page...*

**PXP\_HIST16\_PARAM0 field descriptions (continued)**

Field	Description
15–13 RSVD1	Reserved, always set to zero.
12–8 VALUE1	GRAY1 value for 16-level histogram
7–5 RSVD0	Reserved, always set to zero.
4–0 VALUE0	GRAY0 (Black) value for 16-level histogram

### 38.7.48 16-level Histogram Parameter 1 Register. (PXP\_HIST16\_PARAM1)

This register specifies four of the valid values for a 16-level histogram. If all pixels in a bitmap match the 16-level histogram values, STATUS[3] will be set at the end of frame processing. All comparator values should be programmed such that they are consistent with the PANEL\_MODE control field.

Address: 20F\_0000h base + 2F0h offset = 20F\_02F0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD7			VALUE7					RSVD6			VALUE6				RSVD5			VALUE5				RSVD4			VALUE4						
W																																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	

**PXP\_HIST16\_PARAM1 field descriptions**

Field	Description
31–29 RSVD7	Reserved, always set to zero.
28–24 VALUE7	GRAY7 value for 16-level histogram
23–21 RSVD6	Reserved, always set to zero.
20–16 VALUE6	GRAY6 value for 16-level histogram
15–13 RSVD5	Reserved, always set to zero.

Table continues on the next page...

**PXP\_HIST16\_PARAM1 field descriptions (continued)**

Field	Description
12–8 VALUE5	GRAY5 value for 16-level histogram
7–5 RSVD4	Reserved, always set to zero.
4–0 VALUE4	GRAY4 value for 16-level histogram

### 38.7.49 16-level Histogram Parameter 2 Register. (PXP\_HIST16\_PARAM2)

This register specifies four of the valid values for a 16-level histogram. If all pixels in a bitmap match the 16-level histogram values, STATUS[3] will be set at the end of frame processing. All comparator values should be programmed such that they are consistent with the PANEL\_MODE control field.

Address: 20F\_0000h base + 300h offset = 20F\_0300h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	RSVD11			VALUE11						RSVD10			VALUE10				RSVD9			VALUE9					RSVD8			VALUE8					
W																																	
Reset	0	0	0	0	1	0	1	1	0	0	0	0	1	0	1	0	0	0	0	0	1	0	0	0	1	0	0	0	0	1	0	0	0

**PXP\_HIST16\_PARAM2 field descriptions**

Field	Description
31–29 RSVD11	Reserved, always set to zero.
28–24 VALUE11	GRAY11 value for 16-level histogram
23–21 RSVD10	Reserved, always set to zero.
20–16 VALUE10	GRAY10 value for 16-level histogram
15–13 RSVD9	Reserved, always set to zero.
12–8 VALUE9	GRAY9 value for 16-level histogram

*Table continues on the next page...*

**PXP\_HIST16\_PARAM2 field descriptions (continued)**

Field	Description
7–5 RSVD8	Reserved, always set to zero.
4–0 VALUE8	GRAY8 value for 16-level histogram

**38.7.50 16-level Histogram Parameter 3 Register.  
(PXP\_HIST16\_PARAM3)**

This register specifies four of the valid values for a 16-level histogram. If all pixels in a bitmap match the 16-level histogram values, STATUS[3] will be set at the end of frame processing. All comparator values should be programmed such that they are consistent with the PANEL\_MODE control field.

Address: 20F\_0000h base + 310h offset = 20F\_0310h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD15			VALUE15					RSVD14			VALUE14				RSVD13			VALUE13				RSVD12			VALUE12						
W																																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	0	0	0	0	1	1	0	1	0	0	0	0	1	1	0	0	

**PXP\_HIST16\_PARAM3 field descriptions**

Field	Description
31–29 RSVD15	Reserved, always set to zero.
28–24 VALUE15	GRAY15 (White) value for 16-level histogram
23–21 RSVD14	Reserved, always set to zero.
20–16 VALUE14	GRAY14 value for 16-level histogram
15–13 RSVD13	Reserved, always set to zero.
12–8 VALUE13	GRAY13 value for 16-level histogram
7–5 RSVD12	Reserved, always set to zero.
4–0 VALUE12	GRAY12 value for 16-level histogram



### 38.7.51 PXP Power Control Register. (PXP\_POWER)

Address: 20F\_0000h base + 320h offset = 20F\_0320h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CTRL																ROT_MEM_LP_STATE		LUT_LP_STATE_WAY1_BANKN		LUT_LP_STATE_WAY0_BANKN		LUT_LP_STATE_WAY0_BANK0									
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### PXP\_POWER field descriptions

Field	Description
31–12 CTRL	This register contains power control for the PXP.
11–9 ROT_MEM_LP_STATE	Select the low power state of the ROT memory. 0x0 <b>NONE</b> — Memory is not in low power state. 0x1 <b>LS</b> — Light Sleep Mode. Low leakage mode, maintain memory contents. 0x2 <b>DS</b> — Deep Sleep Mode. Low leakage mode, maintain memory contents. 0x4 <b>SD</b> — Shut Down Mode. Shut Down periphery and core, no memory retention.
8–6 LUT_LP_STATE_WAY1_BANKN	Select the low power state of the LUT's WAY0-BANK0,1,2,3 memory. 0x0 <b>NONE</b> — Memory is not in low power state. 0x1 <b>LS</b> — Light Sleep Mode. Low leakage mode, maintain memory contents. 0x2 <b>DS</b> — Deep Sleep Mode. Low leakage mode, maintain memory contents. 0x4 <b>SD</b> — Shut Down Mode. Shut Down periphery and core, no memory retention.
5–3 LUT_LP_STATE_WAY0_BANKN	Select the low power state of the LUT's WAY0-BANK1,2,3 memory. 0x0 <b>NONE</b> — Memory is not in low power state. 0x1 <b>LS</b> — Light Sleep Mode. Low leakage mode, maintain memory contents. 0x2 <b>DS</b> — Deep Sleep Mode. Low leakage mode, maintain memory contents. 0x4 <b>SD</b> — Shut Down Mode. Shut Down periphery and core, no memory retention.
2–0 LUT_LP_STATE_WAY0_BANK0	Select the low power state of the LUT's WAY0-BANK0 memory. 0x0 <b>NONE</b> — Memory is not in low power state. 0x1 <b>LS</b> — Light Sleep Mode. Low leakage mode, maintain memory contents. 0x2 <b>DS</b> — Deep Sleep Mode. Low leakage mode, maintain memory contents. 0x4 <b>SD</b> — Shut Down Mode. Shut Down periphery and core, no memory retention.

### 38.7.52 Next Frame Pointer (PXP\_NEXT)

This register contains a pointer to a data structure used to reload the PXP registers at the end of the current frame.

To enable this functionality, software must write this register while the PXP is processing the current data frame (if the PXP is currently idle, this will also initiate an immediate load of registers from the pointer). The process of writing this register (WRITE operation) will set a semaphore in hardware to notify the control logic that a register reload operation must be performed when the current frame processing is complete. At the end of a frame, the PXP will fetch the register settings from this location, signal an interrupt to software, then proceed with rendering the next frame of data. Software may cancel the reload operation by issuing a CLEAR operation to this register. SET and TOGGLE operations should not be used when addressing this register. All registers will be reloaded with the exception of the following: STAT, CSCCOEFn, NEXT. All other registers will be loaded in the order they appear in the register map. Once the pointer's contents have been loaded into the PXP's registers, the NEXT\_IRQ interrupt will be issued (see the PXP\_STATUS register).

#### EXAMPLE

```
// create register command structure in memory
u32* pxp_commands0[48], pxp_commands1;
u32 rc;

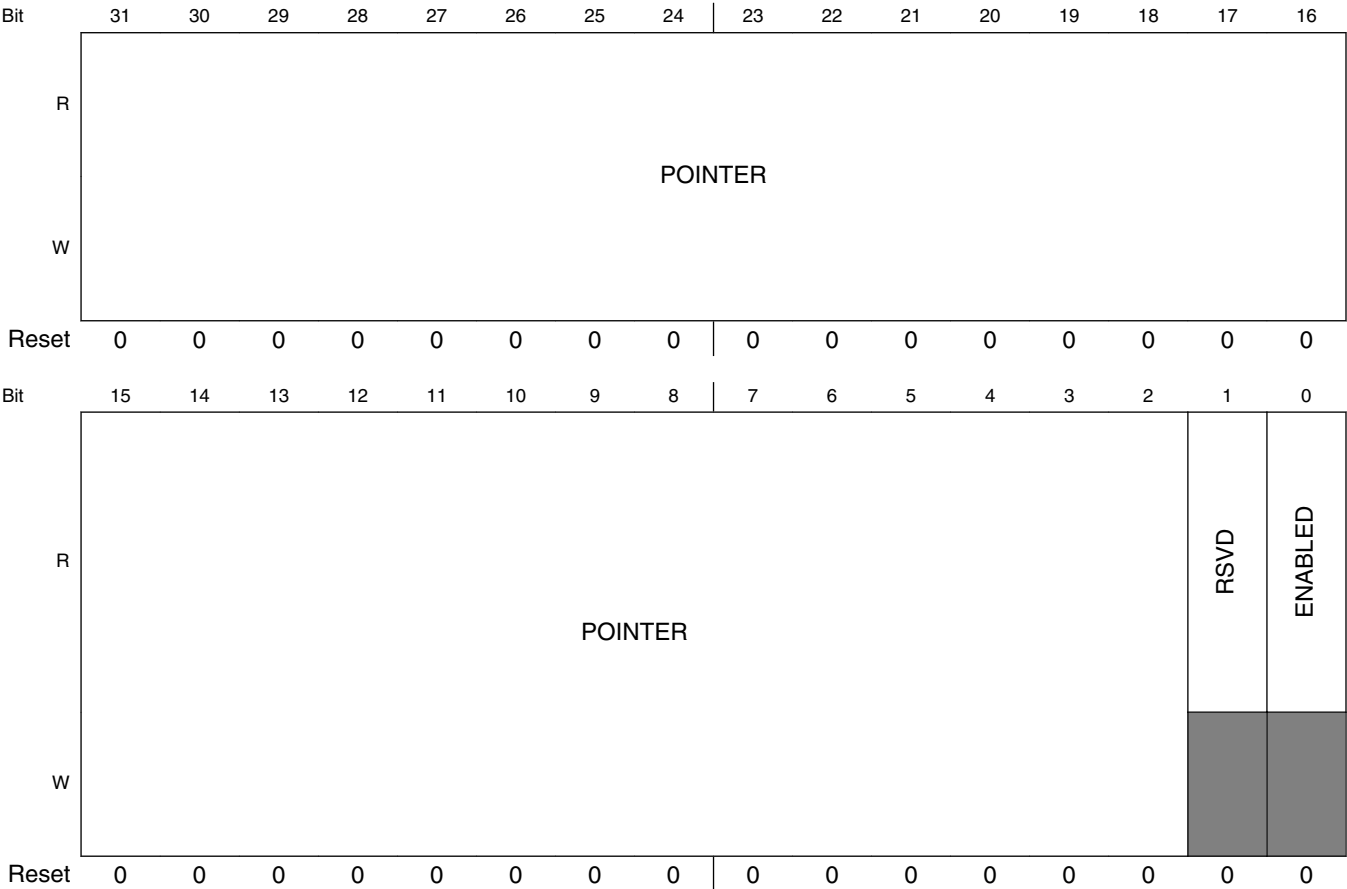
// initialize control structure for frame 0
pxp_commands0[0] = ...; // CTRL
pxp_commands0[1] = ...; // OUT Buffer
...
pxp_commands0[47] = ...; // Overlay7 param2

// initialize control structure for frame 1
pxp_commands1[0] = ...; // CTRL
pxp_commands1[1] = ...; // OUT Buffer
...
pxp_commands1[47] = ...; // Overlay7 param2

// poll until a command isn't queued
while (rc=PXP_NEXT_RD() & BM_PXP_NEXT_ENABLED );
PXP_NEXT_WR(pxp_commands0); // enable PXP operation 0 via command pointer

// poll until first command clears
while (rc=PXP_NEXT_RD() & BM_PXP_NEXT_ENABLED );
PXP_NEXT_WR(pxp_commands1); // enable PXP operation 1 via command pointer
```

Address: 20F\_0000h base + 400h offset = 20F\_0400h



PXP\_NEXT field descriptions

Field	Description
31–2 POINTER	A pointer to a data structure containing register values to be used when processing the next frame. The pointer must be 32-bit aligned and should reside in on-chip or off-chip memory.
1 RSVD	Reserved, always set to zero.
0 ENABLED	Indicates that the "next frame" functionality has been enabled. This bit reflects the status of the hardware semaphore indicating that a reload operation is pending at the end of the current frame.



# Chapter 39

## Random Number Generator (RNGB)

### 39.1 Introduction

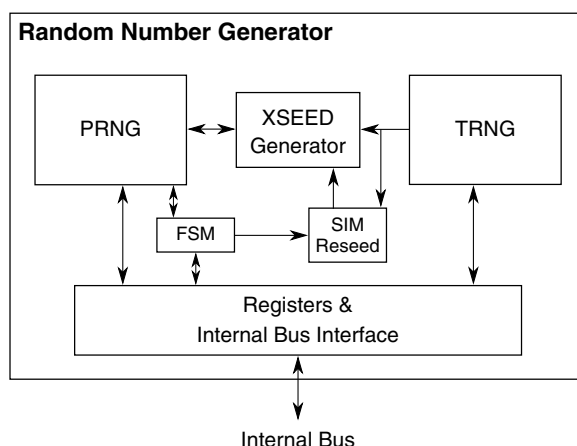
The purpose of the RNGB is to generate cryptographically strong random data.

It uses a true random number generator (TRNG) and a pseudo-random number generator (PRNG) to achieve true randomness and cryptographic strength. The RNGB generates random numbers for secret keys, per message secrets, random challenges, and other similar quantities used in cryptographic algorithms.

This chapter describes the random number generator (RNGB), including a programming model, functional description, and application information.

#### 39.1.1 Block Diagram

The below figure shows the RNGB's three main blocks: PRNG, TRNG, and XSEED generator.



**Figure 39-1. RNG Block Diagram**

### 39.1.2 Features

The RNG includes these distinctive features:

- National Institute of Standards and Technology (NIST)-approved pseudo-random number generator
  - <http://csrc.nist.gov>
- Supports the key generation algorithm defined in the Digital Signature Standard
  - <http://www.itl.nist.gov/fipspubs/fip186.htm>
- Integrated entropy sources capable of providing the PRNG with entropy for its seed.

## 39.2 Modes of Operation

Operational modes of the RNGB can be found here.

### 39.2.1 Self Test Mode

In this mode the RNGB performs a self test of the statistical counters and the PRNG algorithm to verify that the hardware is functioning properly. The self test takes ~29,000 cycles to complete. When self test completes an interrupt may be generated, if there are no outstanding commands in the command register. This mode is entered by setting the RNG\_CMD[ST] bit. When self test mode completes, the RNGB remains idle until seed mode is requested or the RNGB transitions to seed mode if automatic seeding is enabled.

### 39.2.2 Seed Generation Mode

During seed generation, the RNGB adds entropy generated in the TRNG to the 256-bit XKEY register. The PRNG algorithm executes 20,000 times sampling the entropy from the TRNG to create an initial seed for random number generation. At the same time, the TRNG runs simple statistical tests on its output.

When seed generation is complete, the TRNG reports the pass/fail result of the tests through RNG\_ESR. If the new seed passes the statistical tests, RNG\_SR[SDN] is set, signalling that the RNG is ready to compute secure pseudo-random data. The RNG then transitions to random number generation mode.

### 39.2.3 Random Number Generation Mode

When seed generation mode completes and the output FIFO is empty, the RNG enters this mode automatically. Random number generation mode quickly creates computationally random data that is derived by the initial seed produced in seed generation mode.

During random number generation, a new 160-bit random number is generated whenever the five word output FIFO is empty. When the output FIFO contains data, the RNGB automatically enters sleep mode, waiting for the data to be read. When the data is read, the RNGB generates a new 160-bit word and goes back to sleep.

After generating  $2^{20}$  words of random data, the RNGB lets the user know that it requires reseeding through RNG\_SR and continues to generate random data until it is directed to reseed. However, if auto-seeding is selected, the RNGB automatically completes seeding whenever it is needed.

### 39.2.4 Verification Mode

This mode is not a stand-alone mode in that it actually incorporates the previously mentioned three modes: seed generation, entropy generation, and random number generation. When the RNGB is in this mode the three modes behave in a deterministic way. This is done by replacing the TRNG's output with a deterministic sequence. This mode is entered by writing the verification mode bit in the control register.

## 39.3 Memory Map/Register Definition

The following table shows the address map for the RNGB module. Detailed register descriptions are found in the following sections.

**RNG memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
21B_4000	RNGB Version ID Register (RNG_VER)	32	R	1000_0280h	<a href="#">39.3.1/2372</a>
21B_4004	RNGB Command Register (RNG_CMD)	32	R/W	0000_0000h	<a href="#">39.3.2/2373</a>
21B_4008	RNGB Control Register (RNG_CR)	32	R/W	0000_0000h	<a href="#">39.3.3/2374</a>
21B_400C	RNGB Status Register (RNG_SR)	32	R	0000_500Dh	<a href="#">39.3.4/2376</a>
21B_4010	RNGB Error Status Register (RNG_ESR)	32	R	0000_0000h	<a href="#">39.3.5/2378</a>

*Table continues on the next page...*

## RNG memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
21B_4014	RNGB Output FIFO (RNG_OUT)	32	R	0000_0000h	<a href="#">39.3.6/2380</a>
21B_4018	RNGB Entropy Register (RNG_ER)	32	W (always reads 0)	0000_0000h	<a href="#">39.3.7/2380</a>
21B_4020	Verification Control Register (RNG_VCR)	32	R/W	0000_0000h	<a href="#">39.3.8/2381</a>
21B_4024	XKEY Data (RNG_XKEY)	32	R	0010_0000h	<a href="#">39.3.9/2382</a>
21B_4028	Oscillator Counter Control Register (RNG_OCCR)	32	R/W	0000_1000h	<a href="#">39.3.10/2383</a>
21B_402C	Oscillator Counter (RNG_OSC_CNT)	32	R	0000_0000h	<a href="#">39.3.11/2383</a>
21B_4030	Oscillator Counter Status (RNG_OSC_CNT_STAT)	32	R	0000_0000h	<a href="#">39.3.12/2384</a>

## 39.3.1 RNGB Version ID Register (RNG\_VER)

The read-only RNG\_VER register contains the current version of the RNGB. It consists of the RNG type and major and minor revision numbers.

Address: 21B\_4000h base + 0h offset = 21B\_4000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TYPE				0												MAJOR				MINOR											
W																																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0	0	0	0	0	0

## RNG\_VER field descriptions

Field	Description
31–28 TYPE	Random number generator type  0000 RNGA 0001 RNGB (This is the type used in this module) 0010 RNGC Else Reserved
27–16 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
15–8 MAJOR	Major version number.  This field is always set to 0x02.
7–0 MINOR	Minor version number.  Subject to change.



### 39.3.2 RNGB Command Register (RNG\_CMD)

RNG\_CMD controls the RNG's operating modes and interrupt status.

Address: 21B\_4000h base + 4h offset = 21B\_4004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0									0	0	0	0			
W										SR	CE	CI			GS	ST
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**RNG\_CMD field descriptions**

Field	Description
31–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
6 SR	Software reset.  Performs a software reset of the RNGB. This bit is self-clearing.  0 Do not perform a software reset. 1 Software reset.
5 CE	Clear error.  Clears the errors in the RNG_ESR register and the RNGB interrupt. This bit is self-clearing.  0 Do not clear errors and interrupt. 1 Clear errors and interrupt.
4 CI	Clear interrupt.  Clears the RNGB interrupt if an error is not present. This bit is self-clearing.  0 Do not clear interrupt. 1 Clear interrupt.
3–2 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
1 GS	Generate seed.  Initiates the seed generation process. Seed generation starts <ul style="list-style-type: none"> <li>When RNG_SR[BUSY] is cleared</li> <li>If set simultaneously with ST, after self-test</li> </ul> When the seed generation process completes, this bit automatically clears and an interrupt may be generated if all requested operations are complete.

*Table continues on the next page...*

**RNG\_CMD field descriptions (continued)**

Field	Description
	0 Not in seed generation mode. 1 Generate seed mode.
0 ST	Self test.  Initiates a self test of the RNGB's internal logic. The self-test starts <ul style="list-style-type: none"> <li>• When RNG_SR[BUSY] is cleared, or</li> <li>• If set simultaneously with GS, self test takes precedence and is completed first.</li> </ul> When self test completes, this bit automatically clears and an interrupt may be generated if all requested operations are complete.  0 Not in self test mode. 1 Self test mode.

**39.3.3 RNGB Control Register (RNG\_CR)**

Through use of this register, the RNGB can be programmed to provide slightly different functionality based on its desired use.

Address: 21B\_4000h base + 8h offset = 21B\_4008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0				0				MASKERR		MASKDONE		AR		0	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**RNG\_CR field descriptions**

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9–7 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

*Table continues on the next page...*

## RNG\_CR field descriptions (continued)

Field	Description
6 MASKERR	<p>Mask error interrupt.</p> <p>Masks interrupts generated by errors in the RNGB. These errors can still be viewed in RNG_ESR.</p> <p><b>NOTE:</b> Since masked errors do not interrupt the operation of the RNGB and thus hide potentially fatal errors or conditions that could result in corrupted results, it is strongly recommended that errors only be masked while debugging. All errors are considered fatal, requiring the RNGB to be reset. Until the a reset occurs, the RNGB does not service any random data.</p> <p>0 No mask applied. 1 Mask applied to the error interrupt.</p>
5 MASKDONE	<p>Mask done interrupt.</p> <p>Masks interrupts generated upon completion of seed and self test modes. The status of these jobs can be viewed by:</p> <ul style="list-style-type: none"> <li>• Reading RNG_SR and viewing the seed done and self test done bits (RNG_SR[SDN, STDN])</li> <li>• Viewing RNG_CMD for generate seed or self test bits (RNG_CMD[GS,ST]) being set, indicating that the operation is still taking place.</li> </ul> <p>0 No mask applied. 1 Mask applied.</p>
4 AR	<p>Auto-reseed.</p> <p>Setting this bit allows the RNGB to automatically generate a new seed whenever one is needed. This allows software to never use the RNG_CMD[GS], although it is still possible. A new seed is needed whenever the RNG_SR[RS] is set.</p> <p>0 Do not enable automatic reseeding. 1 Enable automatic reseeding.</p>
3–2 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
1–0 FUFMOD	<p>FIFO underflow response mode.</p> <p>Controls the RNGB's response to a FIFO underflow condition.</p> <p>00 Return all zeros and set RNG_ESR[FUFE] 01 Return all zeros and set RNG_ESR[FUFE] 10 Generate bus transfer error 11 Generate interrupt and return all zeros (Overrides RNG_CR[MASKERR])</p>

### 39.3.4 RNGB Status Register (RNG\_SR)

The RNGBSR is a read-only register which reflects the internal status of the RNGB.

Address: 21B\_4000h base + Ch offset = 21B\_400Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	STATPF								ST_PF			0				ERR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	FIFO_SIZE				FIFO_LVL				0	NSDN	SDN	STDN	RS	SLP	BUSY	1
W																
Reset	0	1	0	1	0	0	0	0	0	0	0	0	1	1	0	1

**RNG\_SR field descriptions**

Field	Description
31–24 STATPF	<p>Statistics test pass fail.</p> <p>Indicates pass or fail status of the various statistics tests on the last seed generated.</p> <ul style="list-style-type: none"> <li>• Bit 31 - Long run test (&gt;34)</li> <li>• Bit 30 - Length 6+ run test</li> <li>• Bit 29 - Length 5 run test</li> <li>• Bit 28 - Length 4 run test</li> <li>• Bit 27 - Length 3 run test</li> <li>• Bit 26 - Length 2 run test</li> <li>• Bit 25 - Length 1 run test</li> <li>• Bit 24 - Monobit test</li> </ul> <p>0 Pass. 1 Fail.</p>
23–21 ST_PF	<p>Self Test Pass Fail.</p> <p>Indicates Pass or Fail status of the TRNG, PRNG, and RESEED self tests,</p> <ul style="list-style-type: none"> <li>• Bit 23 - TRNG self test pass/fail</li> </ul>

*Table continues on the next page...*

**RNG\_SR field descriptions (continued)**

Field	Description
	<ul style="list-style-type: none"> <li>• Bit 22 - PRNG self test pass/fail</li> <li>• Bit 21 - RESEED self test pass/fail</li> </ul> <p>0 Pass.</p> <p>1 Fail.</p>
20–17 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
16 ERR	<p>Error.</p> <p>Indicates an error was detected in the RNGB. Read the RNG_ESR register for details.</p> <p>0 No error.</p> <p>1 Error detected.</p>
15–12 FIFO_SIZE	<p>FIFO size.</p> <p>Size of the FIFO, and maximum possible FIFO level. The bits should be interpreted as an integer. This value is set to five on the default version of RNGB.</p>
11–8 FIFO_LVL	<p>FIFO level.</p> <p>Indicates the number of random words currently in the output FIFO. The bits should be interpreted as an integer.</p>
7 Reserved	<p>This field is reserved.</p> <p>This read-only field is reserved and always has the value 0.</p>
6 NSDN	<p>New seed done.</p> <p>Indicates that a new seed is ready for use during the next seed generation process.</p>
5 SDN	<p>Seed done.</p> <p>Indicates the RNG has generated the first seed.</p> <p>0 Seed generation process not complete.</p> <p>1 Completed seed generation since the last reset.</p>
4 STDN	<p>Self test done.</p> <p>Indicates the self test is complete. This bit is cleared by hardware reset or a new self test is initiated by setting RNG_CMD[ST].</p> <p>0 Self test not complete.</p> <p>1 Completed a self test since the last reset.</p>
3 RS	<p>Reseed needed.</p> <p>Indicates the RNGB needs to be reseeded. This is done by setting RNG_CMD[GS], or automatically if RNG_CR[AR] is set.</p> <p>0 RNGB does not need to be reseeded.</p> <p>1 RNGB needs to be reseeded.</p>
2 SLP	<p>Sleep.</p> <p>Indicates if the RNGB is in sleep mode. When set, the RNGB is in sleep mode and all internal clocks are disabled. While in this mode, access to the FIFO is allowed. Once the FIFO is empty, the RNGB fills the FIFO and then enters sleep mode again.</p>

*Table continues on the next page...*

**RNG\_SR field descriptions (continued)**

Field	Description
	0 RNGB is not in sleep mode. 1 RNGB is in sleep mode.
1 BUSY	Busy.  Reflects the current state of RNGB. If RNGB is currently seeding, generating the next seed, creating a new random number, or performing a self test, this bit is set.  0 Not busy. 1 Busy.
0 Reserved	This field is reserved. This read-only field is reserved and always has the value 1.

**39.3.5 RNGB Error Status Register (RNG\_ESR)**

Address: 21B\_4000h base + 10h offset = 21B\_4010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								FUFE		SATE	STE	OSCE	LFE		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**RNG\_ESR field descriptions**

Field	Description
31–5 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.

*Table continues on the next page...*

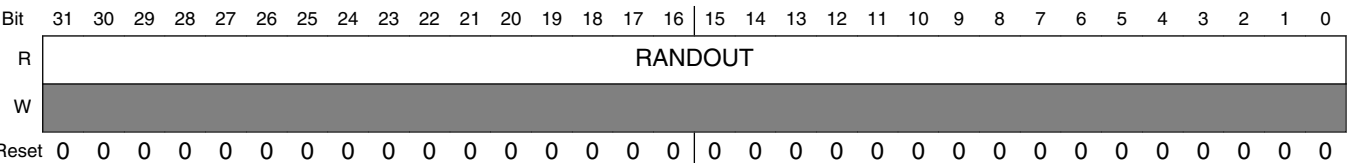
**RNG\_ESR field descriptions (continued)**

<b>Field</b>	<b>Description</b>
4 FUFE	<p>FIFO underflow error</p> <p>Indicates the RNGB has experienced a FIFO underflow condition resulting in the last random data read being unreliable. This bit can be masked by RNG_CR[FUFMOD] and is cleared by hardware or software reset or by writing one to RNG_CMD[CE].</p> <p>0 FIFO underflow has not occurred. 1 FIFO underflow has occurred</p>
3 SATE	<p>Statistical test error.</p> <p>Indicates if RNGB has failed the statistical tests for the last generated seed. This bit is sticky and is cleared by a hardware or software reset or by writing one to RNG_CMD[CE].</p> <p>0 RNGB has not failed the statistical tests. 1 RNGB has failed the statistical tests during initialization.</p>
2 STE	<p>Self test error.</p> <p>Indicates the RNGB has failed the most recent self test. This bit is sticky and can only be reset by a hardware reset or by writing one to RNG_CMD[CE].</p> <p>0 RNGB has not failed self test. 1 RNGB has failed self test.</p>
1 OSCE	<p>Oscillator error.</p> <p>Indicates the oscillator in the RNG may be broken. This bit is sticky and can only be cleared by a software or hardware reset.</p> <p>0 RNG oscillator is working properly. 1 Problem detected with the RNG oscillator.</p>
0 LFE	<p>Linear feedback shift register (LFSR) error.</p> <p>When this bit is set, the interrupt generated was caused by a failure of one of the LFSRs in one of the RNGB's three entropy sources. This bit is sticky and can only be cleared by a software or hardware reset.</p> <p>0 LFSRs are working properly. 1 LFSR failure has occurred.</p>

### 39.3.6 RNGB Output FIFO (RNG\_OUT)

The RNGBOU provides temporary storage for random data generated by the RNGB. This allows the user to read multiple random longwords back-to-back. A read of this address when the FIFO is not empty, returns 32 bits of random data. If the FIFO is read when empty, a FIFO underrun response is returned according to RNG\_CR[FUFMOD]. For optimal system performance, poll RNG\_SR[FIFO\_LVL] to ensure random values are present before reading the FIFO.

Address: 21B\_4000h base + 14h offset = 21B\_4014h



RNG\_OUT field descriptions

Field	Description
31–0 RANDOUT	Random Output

### 39.3.7 RNGB Entropy Register (RNG\_ER)

This read-only register can only be accessed when the RNGB is in Control Access Mode. The RNGBER is a write-only register which allows the user to insert entropy into the RNGB. When written this register causes an addition of the write data to the XKEY within the RNGB. If the system has a means to collect quality entropy (user key strokes or other random patterns), this is the way to add it directly into the accumulated entropy within the RNGB. Writing the RNG\_ER does not have a detrimental effect on the quality of random numbers as the number written is added to the state, not used to replace the existing state. Writing all zeros to the entropy register does nothing to the quality of random numbers generated.

**NOTE**

This register can only be written when the RNG is not busy (RNG\_SR[BUSY] = 0).



Address: 21B\_4000h base + 18h offset = 21B\_4018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																															
W	ENT																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### RNG\_ER field descriptions

Field	Description
31–0 ENT	Entropy input.  This value is added directly to the internal state of the RNGB thus modifying its internal state and affecting future random numbers generated by the RNGB.

## 39.3.8 Verification Control Register (RNG\_VCR)

This register cannot be accessed when RNGB is in Control Access Mode. Through use of this register, the RNGB can be placed in a deterministic mode allowing verification of the design.

Address: 21B\_4000h base + 20h offset = 21B\_4020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						0	0	0							
W							RST_XKEY	RST_SHREG					FAKE_SEED	OSC_TEST	FRC_SYS_CLK	SH_CLK_OFF
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**RNG\_VCR field descriptions**

Field	Description
31–10 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
9 RST_XKEY	Reset XKEY Register. Reset the internal state of the RNGB.
8 RST_SHREG	Reset Shift Registers. Reset the internal LFSRs to a known state.
7–4 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
3 FAKE_SEED	Fake Seed. This allows for quicker structural simulation by faking the seed process, which takes 2000000 cycles.
2 OSC_TEST	Oscillator Frequency Test. This can be set to enable the oscillator for oscillator frequency testing.
1 FRC_SYS_CLK	Force System Clock. This can be set to force the oscillator shift registers to use the system clock for additional test coverage.
0 SH_CLK_OFF	Shift Clocks Off. This can be set to shut off the shift register clocks.

**39.3.9 XKEY Data (RNG\_XKEY)**

This read-only register can only be accessed when the RNGB is in Control Access Mode. The register is used to read out the internal state of the PRNG one 32-bit word at a time. Since the structure is 256-bits, it takes eight reads to read out the entire contents. If reads of this register are not done in multiples of eight, the next group of random data generated will not correctly reflect the state of the RNGB. This register will only return relevant data when the RNGB is not busy.

Address: 21B\_4000h base + 24h offset = 21B\_4024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	XKEY																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**RNG\_XKEY field descriptions**

Field	Description
31–0 XKEY	32-bit chunk of the 256-bit internal XKEY Data Structure.

### 39.3.10 Oscillator Counter Control Register (RNG\_OCCR)

This register cannot be accessed when the RNGB is in the Secure Mode. The register is used to tell the oscillator test logic how long to count the oscillator clock pulses. The value written to the register is decremented every system clock cycle (ipg\_clk) until it reaches zero. When it reaches zero, the Oscillator Counter register is frozen allowing an outside observer to count the number of oscillator clock pulses in a given amount of time. In turn, the approximate frequency of each oscillator clock can be computed.

Address: 21B\_4000h base + 28h offset = 21B\_4028h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														OCCR																	
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	

#### RNG\_OCCR field descriptions

Field	Description
31–18 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
17–0 OCCR	Reading the number of clock cycles remaining; Writing number of clock cycles during which to count the oscillator clock pulses

### 39.3.11 Oscillator Counter (RNG\_OSC\_CNT)

This register can only be accessed when the RNGB is in Control Access Mode. The register is used to count the number of oscillator pulses received from Oscillator starting from the time the RNGB Oscillator Counter Control Register is written and ending at the time the RNGB Oscillator Counter Control Register reaches zero. The Oscillator Counter resets when the RNGB Oscillator Counter Control Register is written to and stops incrementing when the RNGB Oscillator Counter Control Register reaches zero. This register is clocked asynchronously with respect to the ipg\_clk clock domain. In order to avoid unknown values during simulation and meta stable states in silicon, this register should only be read when the RNGB Oscillator Counter Control Register is zero (i.e. when the register is not counting).

Address: 21B\_4000h base + 2Ch offset = 21B\_402Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0													CLOCK_PULSES																		
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### RNG\_OSC\_CNT field descriptions

Field	Description
31–20 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
19–0 CLOCK_ PULSES	CLOCK PULSES

### 39.3.12 Oscillator Counter Status (RNG\_OSC\_CNT\_STAT)

This register can only be accessed when the RNGB is in the Control Access Mode. The bits in this register signal that each oscillator has generated a minimum of 0x400 clock pulses each. After hardware reset, the oscillator counters count oscillator pulses for 0x1000 system clock cycles. So, immediately after reset, the bits in this register verify that the oscillators are running at least 1/4 the frequency of the system clock. This information may be useful on the tester where deterministic behavior is required.

Address: 21B\_4000h base + 30h offset = 21B\_4030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															OS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### RNG\_OSC\_CNT\_STAT field descriptions

Field	Description
31–1 Reserved	This field is reserved. This read-only field is reserved and always has the value 0.
0 OS	Oscillator Status  Indicates that the clock derived from the oscillator has toggled at least 0x400 times since the last write of the Oscillator Counter Control Register (the act of resetting the RNGB can be considered such a write since the Oscillator Counter Control Register resets to a non zero value).  0 Oscillator has not toggled 0x400 times. 1 Oscillator has toggled 0x400 times.

## 39.4 Functional Description

The RNGB performs two functional operations, seed generation and random number generation.

These operations, described in [Modes of Operation](#), are performed in conjunction with the major functional blocks in the RNGB described below.

### 39.4.1 Pseudorandom Number Generator (PRNG)

The PRNG implements the NIST-approved PRNG described in the *Digital Signature Standard*. The 160-bit output of the SHA-1 block is the next five words of random data. The PRNG is designed to generate  $2^{20}$  words of random data before requiring reseeding, using the TRNG only during the seeding/initialization process. The initial seed takes approximately two million clock cycles. After this the RNGB can generate five 32-bit words every 112 clock cycles. Reseeding takes place transparently through use of the simultaneous reseed LFSRs. The entropy stored in this 128-bit LFSR and 128-bit shift register is added directly into the XKEY structure via the RNGB XSEED generator whenever reseeding is required.

### 39.4.2 True Random Number Generator (TRNG)

The TRNG is comprised of two entropy sources each providing a single bit of output. Concatenated together, these two output bits are expected to provide one bit of entropy every 100 clock cycles. In addition to generating entropy, the TRNG also performs several statistical tests on its output. The pass/fail status of these tests are reflected in RNG\_ESR.

### 39.4.3 Resets

There are two ways to reset the RNGB: power-on/hardware reset and software reset. The software reset is functionally equivalent to the power-on/hardware reset. The power-on/hardware reset is asynchronous. Software reset is performed by setting the RNG\_CMD[SR] bit. These are summarized in the table below.

**Table 39-14. Reset Summary**

Reset	Source	Characteristics	Internally resets:	Affect on External Signal:
Hardware	ipg_hard_async_reset_b	Active-low, asynchronous, minimum 1-cycle	All interface registers and puts RNGB into the IDLE state	—
Software	RNG_CMD[SR]	Active-high	All interface registers and puts RNGB into the IDLE state	—

### 39.4.3.1 Power-on/Hardware Reset

Asserting the ipg\_hard\_async\_reset\_b signal sets all interface registers to their default state and puts the state machine into the IDLE mode.

### 39.4.3.2 Software Reset

The software reset is functionally equivalent to the hardware reset, but allows the RNGB to be fully reset by writing to the SW\_RST bit (bit-6) in the RNGB Command Register. This bit is self-resetting. A software reset may be performed at any time.

## 39.4.4 RNG Interrupts

There is a single RNG interrupt generated to the processor's interrupt controller. The source of the interrupt is determined by reading the RNG status register. If an error is the cause of the interrupt, further information is available by reading the RNG error status register. The interrupts can be masked by the RNG\_CR[MASKDONE or MASKERR] bits

It is strongly recommended that the error interrupt is only masked while debugging, since masking the error interrupt could hide potentially fatal errors or conditions that could result in corrupted results. All errors are considered fatal, requiring the RNG to be reset. The RNG does not service any random data until a reset occurs.

The available interrupt sources are described in the following table.

**Table 39-15. RNG Interrupt Sources**

Sources	Status Bit Field	RNG_CR Mask Bit Field	Description
Seed generation done	RNG_SR[SDN]	MASKDONE	First seed was generated
Self test done	RNG_SR[STDN]	MASKDONE	Self test finished
Error	RNG_SR[ERR]	MASKERR	Error detected. See RNG_ESR for details.
Linear feedback shift register (LFSR)	RNG_ESR[LSFRE]	MASKERR	Fault in one of the TRNG's LFSRs
Oscillator	RNG_ESR[OSCE]	MASKERR	TRNG ring oscillator may be malfunctioning
Self test	RNG_ESR[STE]	MASKERR	Self test failed
Statistical test	RNG_ESR[SATE]	MASKERR	Statistics test for last seed generation failed
FIFO Underflow	RNG_ESR[FUFE]	MASKERR	FIFO read while empty

## 39.5 Initialization/Application Information

Information found here describes the module initialization.

### 39.5.1 Manual Seeding

The intended general operation of the RNGB is as follows:

1. Reset/initialize.
2. Write to the RNG\_CR to setup the RNGB for the desired functionality.
3. Write to RNG\_CMD to run self-test or seed generation.
4. Wait for interrupt to indicate completion of the requested operation(s).
5. Repeat steps 3–4 if seed generation is not complete.
6. Poll RNG\_SR for FIFO level.
7. Read available random data from output FIFO.
8. Repeat steps 6 and 7 as needed, until  $2^{20}$  words have been generated.
9. Write to RNG\_CMD to run seed mode.
10. Repeat steps 4–9.

### 39.5.2 Automatic Seeding

The intended general operation of the RNGB with automatic seeding enabled is as follows:

1. Reset/initialize.
2. Write to the RNG\_CR to setup the RNGB for automatic seeding and the desired functionality.
3. Wait for interrupt to indicate completion of first seed
4. Poll RNG\_SR for FIFO level.
5. Read available random data from output FIFO.
6. Repeat steps 4 and 5 as needed. Automatic seeding occurs when necessary and is transparent to operation.

### 39.5.3 Deterministic Mode

This section provides a simple example of how to use the RNGB in deterministic mode. For this process to work, the system must be in a non-secure state.

1. Reset / Initialize RNGB
2. Write to Control Register (bit-8) to Enable Deterministic Mode (System Must not be Secure)
3. Follow steps 2-5 of [Manual Seeding](#) or 2-7 of [Automatic Seeding](#)

### 39.5.4 Oscillator Frequency Verification

This section provides a simple example of how to use the RNGB in deterministic mode. This test will allow an approximate determination of the oscillator frequency. In order for this process to work, the system must be in a non-secure state.

1. Reset / Initialize RNGB
2. Write to Control Register (bit-9) to Enable Control Access (System Must not be in the Secure State)



3. Write to the Verification Control Register (bit-2) to enable the TRNG Oscillator Frequency Test
4. Write to the Oscillator Counter Control Register, the number of system clocks (Nsys), required for the test
5. Poll the Oscillator Counter Control Register till 0 is read.
6. Read the number of oscillator clock cycles (Nosc) from the Oscillator Counter Register. From the system clock frequency (fsys), the number of system clocks used and the number of oscillator clock cycles read (Nosc) an approximate frequency (fosc) of the TRNG oscillator can be calculated as follows:
  - a. Test period =  $T_p$
  - b.  $T_p = N_{sys} / f_{sys}$
  - c. TRNG oscillator period =  $T_{osc}$
  - d.  $T_{osc} = T_p / N_{osc}$
  - e. TRNG Oscillator frequency =  $1 / T_{osc}$

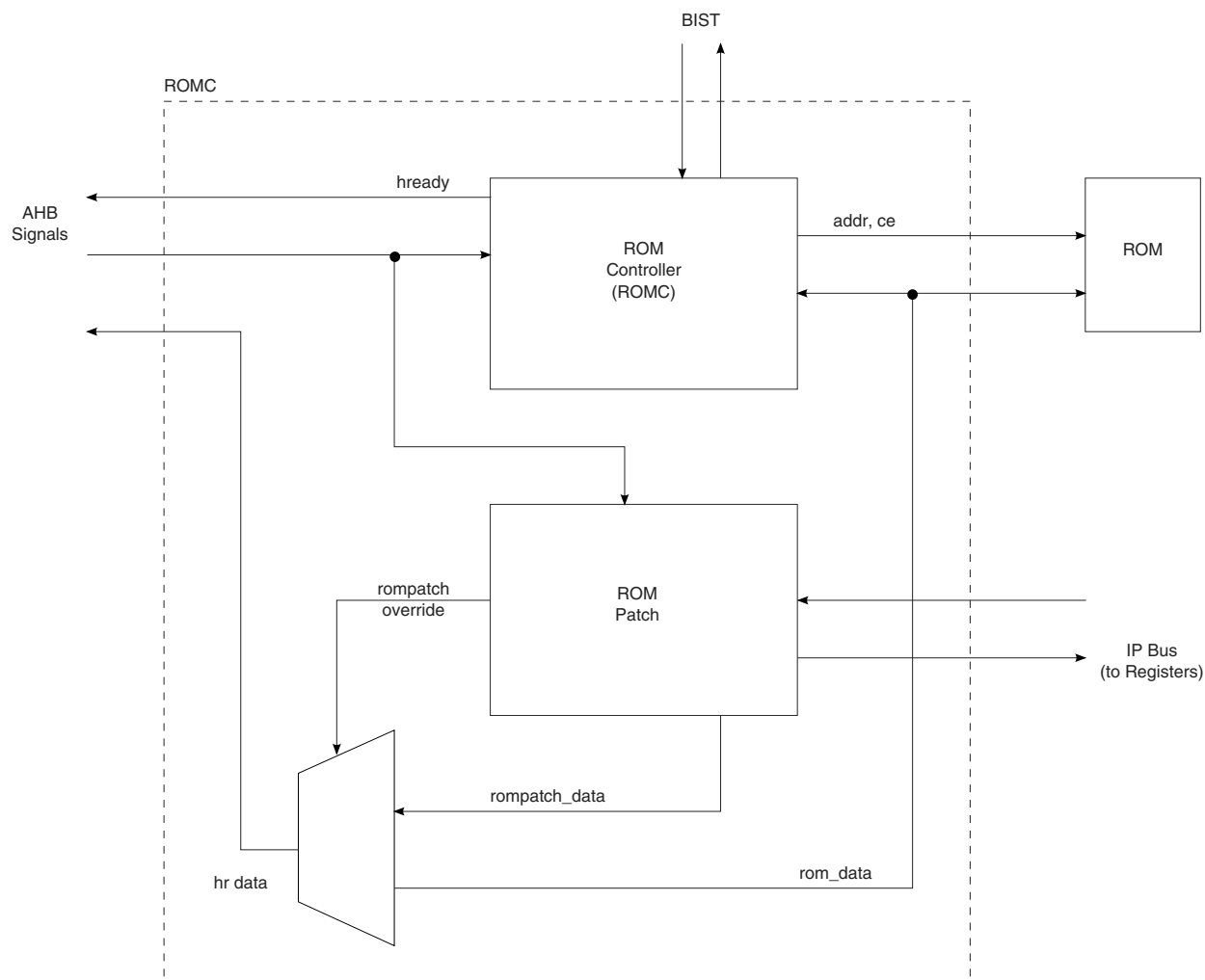


## **Chapter 40**

# **ROM Controller with Patch (ROMC)**

### **40.1 Overview**

The Read Only Memory Controller with ROM Patch (ROMC) acts as an interface between the ARM advanced high-performance bus (AHB - Lite) and the Read Only Memory. The ROMC consists of a ROM Controller and a ROM Patch. The ROM Patch is used to either patch code routines or fix data tables in the ROM area. There is an IP Bus interface to access the ROM Patch Registers and. The figure below depicts the main functional sub-blocks of the ROMC.



**Figure 40-1. ROMC Block Diagram**

### 40.1.1 Features

- Supports ROM size ranges from 16 Kbyte up to 4 Mbyte with increments of 1 Kbyte
- Supports opcode patching for a maximum of 16 different addresses in 4 Mbytes of ROM space
- Supports one-word data fixes for a max of 8 memory locations in 4 Mbytes of ROM space
- Supports patching of the Reset Vector (at 0x0000\_0000) to allow external booting

### 40.1.2 Modes of Operation

There are two modes of operation: normal mode and BIST mode.

In normal mode (`ipt_bist_en = 0`), the ROMC ensures correct reads from the ROM, assuming the memory complies with the characteristics and requirements for which the ROMC was designed.

### 40.1.2.1 Low Power Mode

There are two clock enables that are used to switch off parts of the ROMC logic when inactive. The first clock enable is used to disable the ROM Controller when the master connected to the AHB interface is not initiating a read to the ROM. The second clock enable is used to disable the registers used to program the ROM patch feature when the registers are not being accessed.

## 40.2 Clocks

The table found here describes the clock sources for ROMCP.

Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 40-1. ROMCP Clocks**

Clock name	Clock Root	Description
hclk	ahb_clk_root	System / bus clock
hclk_reg	ipg_clk_root	System access clock
96krom_CLK	ahb_clk_root	ROM clock

## 40.3 Memory Map

### 40.3.1 ROM Memory Map in detail

The ROMC supports ROM sizes with a range of 16 Kbyte to 4 Mbyte with an increment of 1 Kbyte. The 16 Kbyte lower limit was chosen because the minimum size of security code on an ARM platform is approximately 16 Kbyte of code, which is only accessible in supervisor mode. Note that it is the MMU that controls whether any region of memory is secure.

The exception vectors must be secured as well, and must be put in the same area as the security code. Since they must reside at address 0x0000\_0000, the entire 16 Kbyte of ROM which can only be accessible in supervisor mode is located at the very beginning of the platform memory map.

If the user chooses not to use the security code, a memory size smaller than 16 Kbyte can be connected to the platform (minimum of 1 Kbyte). The MMU can be programmed to allow any kind of access into this memory. However, if the ROM size is less than 16 Kbyte, memory aliasing will occur for all invalid addresses greater than the memory size but within the 16 Kbyte of space.

For ROM sizes bigger than 16 Kbyte, the rest of its physical size resides at the address starting at 0x0040\_4000 (4 M+16 Kbyte) going up to [0x0040\_4000 + (mem. size - 16Kbyte)]. For examples on power-of-two memories please see [Figure 1](#).

## **40.4 Functional Description**

This section is divided up into the ROM Controller Functional Description and the ROMC functional description.

### **40.4.1 ROM Controller (ROMC) Functional Description**

#### **40.4.1.1 Functionality overview**

The ROMC serves two main functions. First, as an interface between the AHB-Lite bus on an ARM platform and the ROM. Second, it drives and receives several signals for the BIST engine. In normal mode of operation, the ROMC monitors the AHB-Lite for memory access requests and performs the memory operation to the ROM.

The ROMC includes the option to wait state all accesses from either the ARM or non-ARM masters to ROM in the event that timing requirements will not allow single hclk clock cycle reads. If a wait state is required, the static inputs rom\_wait\_arm or rom\_wait\_alt\_mstr can be set to 1 and accesses will take two hclk clock cycles. If wait states are not required, rom\_wait\_arm or rom\_wait\_alt\_mstr can be set to 0 and accesses will take one hclk clock cycle to complete.

### **40.4.2 ROMC Functional Description**

### 40.4.2.1 ROMC Disabling

All the bits in the ROMC\_ROMPATCHENL register are cleared on Reset, disabling all the address comparators. Once the comparators have been enabled, the ROMC functions of data fixing and opcode patching can be quickly disabled by setting the DIS bit in the ROMC\_ROMPATCHCNTL register. This bit is used to enable secure operations in which patching functions need to be disabled. This bit is cleared on Reset.

### 40.4.2.2 ROMC Event Priority

The ROMC has a total of 16 address comparators. The first 8 (0 through 7) comparators can be programmed for the data fixing function (through the 8 data fix enable bits in the ROMC\_ROMPATCHCNTL register) while the rest are for opcode patching by default. This allows for potential multiple matching events involving both data fixing and opcode patch types. In these cases the ROMC assigns the highest priority to a data fixing event.

For example, if the ROMC is set up to data fix a certain address with comparator 4 and also opcode patch the same address with comparator 7, it will let comparator 4 have higher priority in indicating a match, and data from ROMC\_ROMPATCHD4 will be put on the rompatch\_romc\_hrdata bus as the override value.

If multiple address matches of the same type level occur concurrently, then the ROMC will choose the source number based on the one with the highest source number. For example, the ROMC is setup to data fix the same location with address comparators 4 and 7, then address comparator 7 will have higher priority in indicating a match, and the value from ROMC\_ROMPATCHD7 will be put on the rompatch\_romc\_hrdata bus as the override value. The same priority applies for an opcode patch event, except the override data is in the form of an SWI instruction with the comment field set to the source number with the highest priority.

### 40.4.2.3 Data Fixing

The data fixing feature allows ROM data to be updated by direct replacement when it is being read. This data usually originates from data tables, but can include ARM instructions. To enable data fixing on a certain address, this address value is written in to one of the first eight (0 through 7) of ROMC\_ROMPATCHAxx registers and the same numbered bit set in the ROMC\_ROMPATCHENL and ROMC\_ROMPATCHCNTL registers. The data to be used for replacement is placed in the corresponding ROMC\_ROMPATCHDxx.

The ROMC looks for a read access to ROM (either code fetch or data load) by snooping the AHB interface for read transactions. The address is compared with the values stored in the ROMC\_ROMPATCHAxx[22:2] registers. If a match occurs from one of the comparators, the ROMC places the value in the corresponding ROMC\_ROMPATCHDxx register on the read data bus by overriding the read data coming from the actual ROM (see the mux in [Figure 40-1](#)). The value on the read data bus is maintained until hready is asserted to terminate the access. In data fixing, the entire word is replaced so if a byte or half-word access occurs on a "data fix" location, the entire data word is replaced. The word being replaced is word aligned. (The two LSBs of the matching ROMC\_ROMPATCHAxx are ignored in the data fix operation.)

#### 40.4.2.4 Opcode Patching

The opcode patch feature provides the ARM core a mechanism to fetch updated versions of code routines that were originally programmed in ROM. This patching mechanism makes use of the SWI (software interrupt instruction) and a table of function pointers residing in writable memory. The opcode being patched is replaced with a SWI instruction by the ROMC. Subsequent processing of the SWI reads from a function pointers table to obtain the address of the replacement code. Execution resumes with this code patch.

To enable opcode patching of a certain address, this address value is written into one of the ROMPATCHAxx registers and the corresponding bit set in the ROMPATCHENL to enable the associated comparator. The register's LSB (ROMC\_ROMPATCHxx[0]) should be set if THUMB mode patching is in effect for this address. The ROMC identifies a ROM read access by snooping the AHB interface. The address is compared with the values stored in the ROMC\_ROMPATCHAxx[22:2] registers. If a match occurs from one of the comparators, the ROMC generates the opcode of a software interrupt (SWI) instruction with the comment field containing the number of the matching address comparator. This opcode and comment is placed on the read data bus until hready is asserted by the ROM controller to terminate the read access.

The type of SWI generated, (that is, either ARM or THUMB), is determined by the LSB of the ROMC\_ROMPATCHAxx register associated with the opcode patch. This bit is cleared for ARM mode (32 bits). The ROMC generates a 32-bit SWI (opcode field is 0xEF, occupying bits [31:24] of the word), with the least significant 5 bits of the 24-bit comment field (bits [23:0]) containing the number of the matching address comparator. The rest of the comment field is filled with zeros. This means that the ROMC will use 16 of the 16777216 possible software interrupts. The ROMC overrides the read data from the ROM.



If the LSB of the matching ROMC\_ROMPATCHAxx register is set, the opcode patch is in THUMB mode (16 bits or half word). The ROMC generates a 16-bit SWI instruction (opcode field is 0xDF, occupying bits [15:8] of the half word) with the least significant 5 bits of the 8-bit comment field containing with the source number of the address comparator. The rest of the comments field is filled with zeros. This means that the ROMC will use 16 of the 256 possible software interrupts. The ROMC puts this 16 bit SWI instruction value on the proper half of the rompatch\_romc\_hrdata bus. The other half is zeroed out. Which half of the bus contains the SWI opcode and comment depends on the mode (Big Endian or Little Endian) and the bit 1 of the matching ROMC\_ROMPATCHAxx register. In Little Endian mode, the lower half is bits {15:0} and the upper half is bits {31:16}. The order is reversed in Big Endian mode.

In Little Endian mode (bigend signal negated), if bit 1 of the matching ROMC\_ROMPATCHAxx is cleared (lower half word selected) then the SWI instruction is put on the lower 16 bits of the read data bus and the upper 16 bits are zeroed out. Only the lower 16 bits of the read data bus is overwritten by the ROMC data. If ROMC\_ROMPATCHAxx[1] is set (upper half word selected), the SWI instruction is put on the upper 16 bits of the read data bus and the lower 16 bits are zeroed out. Only the upper 16 bits of the read data bus is overwritten.

In Big Endian mode (bigend asserted), if bit 1 of the matching ROMC\_ROMPATCHAxx is cleared (lower half word selected) then the SWI instruction is put on the upper 16 bits of the read data bus while the lower 16 bits are zeroed out. Only the upper 16 bits of the read data bus is overwritten. If ROMC\_ROMPATCHAxx[1] is set (upper word selected), the SWI instruction is put on the lower 16 bits and the upper 16 bits are zeroed out. Only the lower 16 bits of the read data bus is overwritten.

The eventual execution of the SWI causes the ARM to save the CPSR in SPSR\_SVC, the address of the next instruction after the SWI in R14\_SVC, enter Supervisor mode, and fetch the SWI vector at 0x8, which then takes it to a handler for further processing as described in the next section.

#### 40.4.2.4.1 Typical Software Response to Opcode Patch

When the SWI handler executes it needs to determine whether the SWI was generated by the ROMC. This is done by loading the SWI instruction and extracting its comment field. The state of the ARM core (ARM or THUMB) when the SWI was executed dictates whether to load the instruction word (ARM) or half word (THUMB). This state information can be determined by testing the T bit (bit 5) of the SPSR. If it's set, the execution was in THUMB mode.

By convention, if the comment field of the SWI is greater than 16, the software interrupt was initiated by software (i.e. an operating system call), and a branch is taken to the appropriate handler routine for further processing. If the comment field is less than 16, the SWI was generated by the ROMC performing a code patch operation. In this case, the software then reads from a table of function pointers, using the value in the SWI comment field as the index into the table. The value that is read is the address of the code patch. This value is loaded into the PC to begin the execution of the code patch. The following code segment illustrates a typical handling of the SWI.

```

stmfd      sp!, {r0-r1,lr}      @ push register onto SWI stack
mrs        r0, spsr             @ get saved status register
tst        r0, #0x20            @ check if call was in THUMB mode
ldrneh     r0, [lr,#-2]         @ yes: load opcode half-word and
bicne      r0, r0, #0xff00      @ yes: extract THUMB comment
ldreq      r0, [lr,#-4]         @ no: load opcode word and
biceq      r0, r0, #0xff000000  @ no: extract ARM comment
                                @ now r0 has comment field
cmp        r0, #16              @ compare to 16 (maximum for ROMC)
ldrlt      lr, =rompatch_tbl_ptr @ < 16: get top of current ROMC
                                @ table; global variable which is
                                @ changeable per context
ldrlt      r1, [lr, r0, lsl #2] @ < 16: read function pointer from
                                @ table assumed an array of pointers
                                @ patch functions
strlt      r1, [sp, #8]         @ < 16: store function pointer onto
                                @ stack in position of link register
ldmpltfd   sp!, {r0-r1,pc}^     @ < 16: "fake" return from SWI, will
                                @ vector core to appropriate patch
                                @ function and set core back to previous
                                @ mode of operating
ldr        r1, =swi_hdlr       @ >= 16: pointer to standard SWI
                                @ handler
mov        lr, pc               @ >= 16: set link register
bx         r1                   @ >= 16: jump to standard SWI
                                @ handler
ldmfd      sp!, {r0-r1,pc}^     @ >= 16: pop registers from stack

```

#### 40.4.2.5 External Boot Feature

Following a Reset event, the ARM issues an instruction fetch of the Reset Vector from address 0x0. This instruction, normally residing in ROM is usually a branch to a Reset handler or boot code which also normally resides in ROM. The ROMC external boot feature allows the bypassing of this code, using a different boot code residing perhaps in external memory.

This feature uses the data fix mechanism and works as follows: if the boot\_int signal is negated when a Reset event occurred, the ROMC will perform a data fix of the Reset Vector at 0x0 with the following instruction (opcode 0xE59FF00C):

```

ldr        pc, [pc, #12]        @ read 0x0000_0014 for reset_vector

```

The value of PC when this instruction is executed is 8 so that a PC relative offset of 12 makes the source address 20 or 0x14. When this instruction executes, the ARM core reads from address 0x0000\_0014, triggering a ROMC data fix operation which places the

value taken from the external boot address on the read data bus, with the two LSBs zeroed out. This value is returned to the ARM to be placed in the PC causing code fetch and execution to start from that address.

#### 40.4.2.6 Alternate Masters and ROMC

The ROMC sits on the AHB bus of the internal ROM (ROMC). This means that the ROMC can modify values on the read data bus going to the master. Therefore, any master which reads an opcode patched or data patched location will read patched data.

### 40.5 ROMCP Memory Map/Register Definition

All registers are accessible through an IP Bus and can only be accessed in privileged mode. These registers can only be written with 32-bits stores and are clocked by hclk\_reg.

The ROMC register placement was originated from the AWPT design used in the ARM7 platform of Neptune

**ROMC memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
21A_C0D4	ROMC Data Registers (ROMC_ROMPATCH0D)	32	R/W	0000_0000h	<a href="#">40.5.1/2400</a>
21A_C0D8	ROMC Data Registers (ROMC_ROMPATCH1D)	32	R/W	0000_0000h	<a href="#">40.5.1/2400</a>
21A_C0DC	ROMC Data Registers (ROMC_ROMPATCH2D)	32	R/W	0000_0000h	<a href="#">40.5.1/2400</a>
21A_C0E0	ROMC Data Registers (ROMC_ROMPATCH3D)	32	R/W	0000_0000h	<a href="#">40.5.1/2400</a>
21A_C0E4	ROMC Data Registers (ROMC_ROMPATCH4D)	32	R/W	0000_0000h	<a href="#">40.5.1/2400</a>
21A_C0E8	ROMC Data Registers (ROMC_ROMPATCH5D)	32	R/W	0000_0000h	<a href="#">40.5.1/2400</a>
21A_C0EC	ROMC Data Registers (ROMC_ROMPATCH6D)	32	R/W	0000_0000h	<a href="#">40.5.1/2400</a>
21A_C0F0	ROMC Data Registers (ROMC_ROMPATCH7D)	32	R/W	0000_0000h	<a href="#">40.5.1/2400</a>
21A_C0F4	ROMC Control Register (ROMC_ROMPATCHCNTL)	32	R/W	0840_0000h	<a href="#">40.5.2/2401</a>
21A_C0F8	ROMC Enable Register High (ROMC_ROMPATCHENH)	32	R	0000_0000h	<a href="#">40.5.3/2402</a>
21A_C0FC	ROMC Enable Register Low (ROMC_ROMPATCHENL)	32	R/W	0000_0000h	<a href="#">40.5.4/2402</a>
21A_C100	ROMC Address Registers (ROMC_ROMPATCH0A)	32	R/W	0000_0000h	<a href="#">40.5.5/2403</a>
21A_C104	ROMC Address Registers (ROMC_ROMPATCH1A)	32	R/W	0000_0000h	<a href="#">40.5.5/2403</a>
21A_C108	ROMC Address Registers (ROMC_ROMPATCH2A)	32	R/W	0000_0000h	<a href="#">40.5.5/2403</a>
21A_C10C	ROMC Address Registers (ROMC_ROMPATCH3A)	32	R/W	0000_0000h	<a href="#">40.5.5/2403</a>

*Table continues on the next page...*

## ROMC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
21A_C110	ROMC Address Registers (ROMC_ROMPATCH4A)	32	R/W	0000_0000h	<a href="#">40.5.5/2403</a>
21A_C114	ROMC Address Registers (ROMC_ROMPATCH5A)	32	R/W	0000_0000h	<a href="#">40.5.5/2403</a>
21A_C118	ROMC Address Registers (ROMC_ROMPATCH6A)	32	R/W	0000_0000h	<a href="#">40.5.5/2403</a>
21A_C11C	ROMC Address Registers (ROMC_ROMPATCH7A)	32	R/W	0000_0000h	<a href="#">40.5.5/2403</a>
21A_C120	ROMC Address Registers (ROMC_ROMPATCH8A)	32	R/W	0000_0000h	<a href="#">40.5.5/2403</a>
21A_C124	ROMC Address Registers (ROMC_ROMPATCH9A)	32	R/W	0000_0000h	<a href="#">40.5.5/2403</a>
21A_C128	ROMC Address Registers (ROMC_ROMPATCH10A)	32	R/W	0000_0000h	<a href="#">40.5.5/2403</a>
21A_C12C	ROMC Address Registers (ROMC_ROMPATCH11A)	32	R/W	0000_0000h	<a href="#">40.5.5/2403</a>
21A_C130	ROMC Address Registers (ROMC_ROMPATCH12A)	32	R/W	0000_0000h	<a href="#">40.5.5/2403</a>
21A_C134	ROMC Address Registers (ROMC_ROMPATCH13A)	32	R/W	0000_0000h	<a href="#">40.5.5/2403</a>
21A_C138	ROMC Address Registers (ROMC_ROMPATCH14A)	32	R/W	0000_0000h	<a href="#">40.5.5/2403</a>
21A_C13C	ROMC Address Registers (ROMC_ROMPATCH15A)	32	R/W	0000_0000h	<a href="#">40.5.5/2403</a>
21A_C208	ROMC Status Register (ROMC_ROMPATCHSR)	32	w1c	0000_0000h	<a href="#">40.5.6/2404</a>

## 40.5.1 ROMC Data Registers (ROMC\_ROMPATCHnD)

The ROMC data registers (ROMC\_ROMPATCHD7 through ROMC\_ROMPATCHD0) store the data to use for the 8 1-word data fix events. Each register is associated with an address comparator (7 through 0). When a data fixing event occurs, the value in the data register corresponding to the comparator that has the address match is put on the romc\_hrdata[31:0] bus until romc\_hready is asserted by the ROM controller to terminate the access. A MUX external to the ROMC will select this data over that of romc\_hrdata[31:0] in returning read data to the ARM core. The selection is done with the control bus rompatch\_romc\_hrdata\_ovr[1:0] with both bits asserted by the ROMC.

If more than one address comparators match, the highest-numbered one takes precedence, and the value in corresponding data register is used for the patching event.

Address: 21A\_C000h base + D4h offset + (4d × i), where i=0d to 7d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATAx																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

## ROMC\_ROMPATCHnD field descriptions

Field	Description
31–0 DATAx	Data Fix Registers - Stores the data used for 1-word data fix operations. The values stored within these registers do not affect the writes to the memory system. They are selected over the read data from ROM when a data fix event occurs.

**ROMC\_ROMPATCHnD field descriptions (continued)**

Field	Description
	If any part of the 1-word data fix is read, then the entire word is replaced. Therefore, a byte or half-word read will cause the ROMC to replace the entire word. The word is word address aligned.

**40.5.2 ROMC Control Register (ROMC\_ROMPATCHCNTL)**

The ROMC control register (ROMC\_ROMPATCHCNTL) contains the block disable bit and the data fix enable bits. The block disable bit provides a means to disable the ROMC data fix and opcode patching functions, even when the address comparators are enabled. The External Boot feature is not affected by this bit. The eight data fix enable bits (0 through 7), when set, assign the associated address comparators to data fix operations

**NOTE**

Bits 27 and 22 always read as 1s.

Address: 21A\_C000h base + F4h offset = 21A\_C0F4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R																
W																
Reset	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

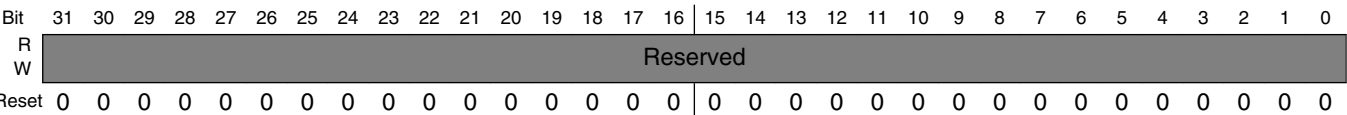
**ROMC\_ROMPATCHCNTL field descriptions**

Field	Description
31–30 -	This field is reserved. Reserved
29 DIS	ROMC Disable -- This bit, when set, disables all ROMC operations. This bit is used to enable secure operations.  0 Does not affect any ROMC functions (default) 1 Disable all ROMC functions: data fixing, and opcode patching
28–8 -	This field is reserved. Reserved
7–0 DATAFIX	<b>Data Fix Enable - Controls the use of the first 8 address comparators for 1-word data fix or for code patch routine.</b>  0 Address comparator triggers a opcode patch 1 Address comparator triggers a data fix

### 40.5.3 ROMC Enable Register High (ROMC\_ROMPATCHENH)

The ROMC enable register high (ROMC\_ROMPATCHENH) and ROMC enable register low (ROMC\_ROMPATCHENL) control whether or not the associated address comparator can trigger a opcode patch or data fix event. This implementation of the ROMC only has 16 comparators, therefore ROMC\_ROMPATCHENH and the upper half of ROMC\_ROMPATCHENL are read-only. ROMC\_ROMPATCHENL[15:0] are associated with comparators 15 through 0. ROMC\_ROMPATCHENLH[31:0] would have been associated with comparators 63 through 32.

Address: 21A\_C000h base + F8h offset = 21A\_C0F8h



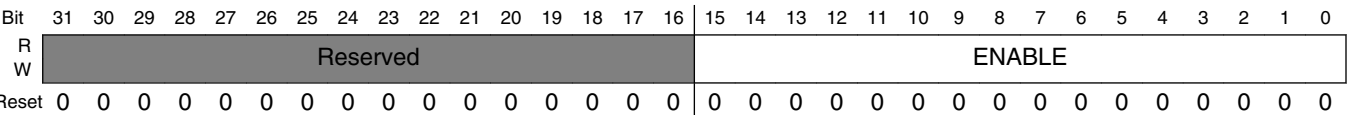
ROMC\_ROMPATCHENH field descriptions

Field	Description
31–0 -	This field is reserved. Reserved

### 40.5.4 ROMC Enable Register Low (ROMC\_ROMPATCHENL)

The ROMC enable register high (ROMC\_ROMPATCHENH) and ROMC enable register low (ROMC\_ROMPATCHENL) control whether or not the associated address comparator can trigger a opcode patch or data fix event. This implementation of the ROMC only has 16 comparators, therefore ROMC\_ROMPATCHENH and the upper half of ROMC\_ROMPATCHENL are read-only. ROMC\_ROMPATCHENL[15:0] are associated with comparators 15 through 0. ROMC\_ROMPATCHENLH[31:0] would have been associated with comparators 63 through 32.

Address: 21A\_C000h base + FCh offset = 21A\_C0FCh



**ROMC\_ROMPATCHENL field descriptions**

Field	Description
31–16 Reserved	This field is reserved.
15–0 ENABLE	<b>Enable Address Comparator</b> - This bit enables the corresponding address comparator to trigger an event.  0 Address comparator disabled 1 Address comparator enabled, ROMC will trigger a opcode patch or data fix event upon matching of the associated address

**40.5.5 ROMC Address Registers (ROMC\_ROMPATCHnA)**

The ROMC address registers (ROMC\_ROMPATCHA0 through ROMC\_ROMPATCHA15) store the memory addresses where opcode patching begins and data fixing occurs. The address registers ROMC\_ROMPATCHA0 through ROMC\_ROMPATCHA15 are each 21 bits wide and dedicated to one 4 Mbyte memory space. Bits 21 through 2 are address bits, to be compared with romc\_haddr[21:2] for a match; bit 1 is also an address bit used for half word selection. Bit 0 is the mode bit (set to 1 for THUMB mode). 1-word data fixing can only be used on the first 8 of the address comparators. ROMC\_ROMPATCHA0 through ROMC\_ROMPATCHA15 are associated each with address comparators 0 through 15.

Address: 21A\_C000h base + 100h offset + (4d × i), where i=0d to 15d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								0	ADDRX						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADDRX															THUMBX
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**ROMC\_ROMPATCHnA field descriptions**

Field	Description
31–23 -	This field is reserved. Reserved

Table continues on the next page...

**ROMC\_ROMPATCHnA field descriptions (continued)**

Field	Description
22–1 ADDRX	Address Comparator Registers - Indicates the memory address to be watched. All 16 registers can be used for code patch address comparison. Only the first 8 registers can be used for a 1-word data fix address comparison.  Bit 1 is ignored if data fix. Only used in code patch
0 THUMBX	THUMB Comparator Select - Indicates that this address will trigger a THUMB opcode patch or an ARM opcode patch. If this watchpoint is selected to be a data fix, then this bit is ignored as all data fixes are 1-word data fixes.  0 ARM patch 1 THUMB patch (ignore if data fix)

**40.5.6 ROMC Status Register (ROMC\_ROMPATCHSR)**

The ROMC status register (ROMC\_ROMPATCHSR) indicates the current state of the ROMC and the source number of the most recent address comparator event.

Address: 21A\_C000h base + 208h offset = 21A\_C208h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved														SW	Reserved
W															w1c	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										SOURCE					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**ROMC\_ROMPATCHSR field descriptions**

Field	Description
31–18 -	This field is reserved. Reserved
17 SW	ROMC AHB Multiple Address Comparator matches Indicator - Indicates that multiple address comparator matches occurred. Writing a 1 to this bit will clear this it.  0 no event or comparator collisions 1 a collision has occurred
16–6 -	This field is reserved. Reserved
5–0 SOURCE	ROMC Source Number - Binary encoding of the number of the address comparator which has an address match in the most recent patch event on ROMC AHB. If multiple matches occurred, the highest priority source number is used.

*Table continues on the next page...*



**ROMC\_ROMPATCHSR field descriptions (continued)**

Field	Description
0	Address Comparator 0 matched
1	Address Comparator 1 matched
15	Address Comparator 15 matched



# Chapter 41

## Smart Direct Memory Access Controller (SDMA)

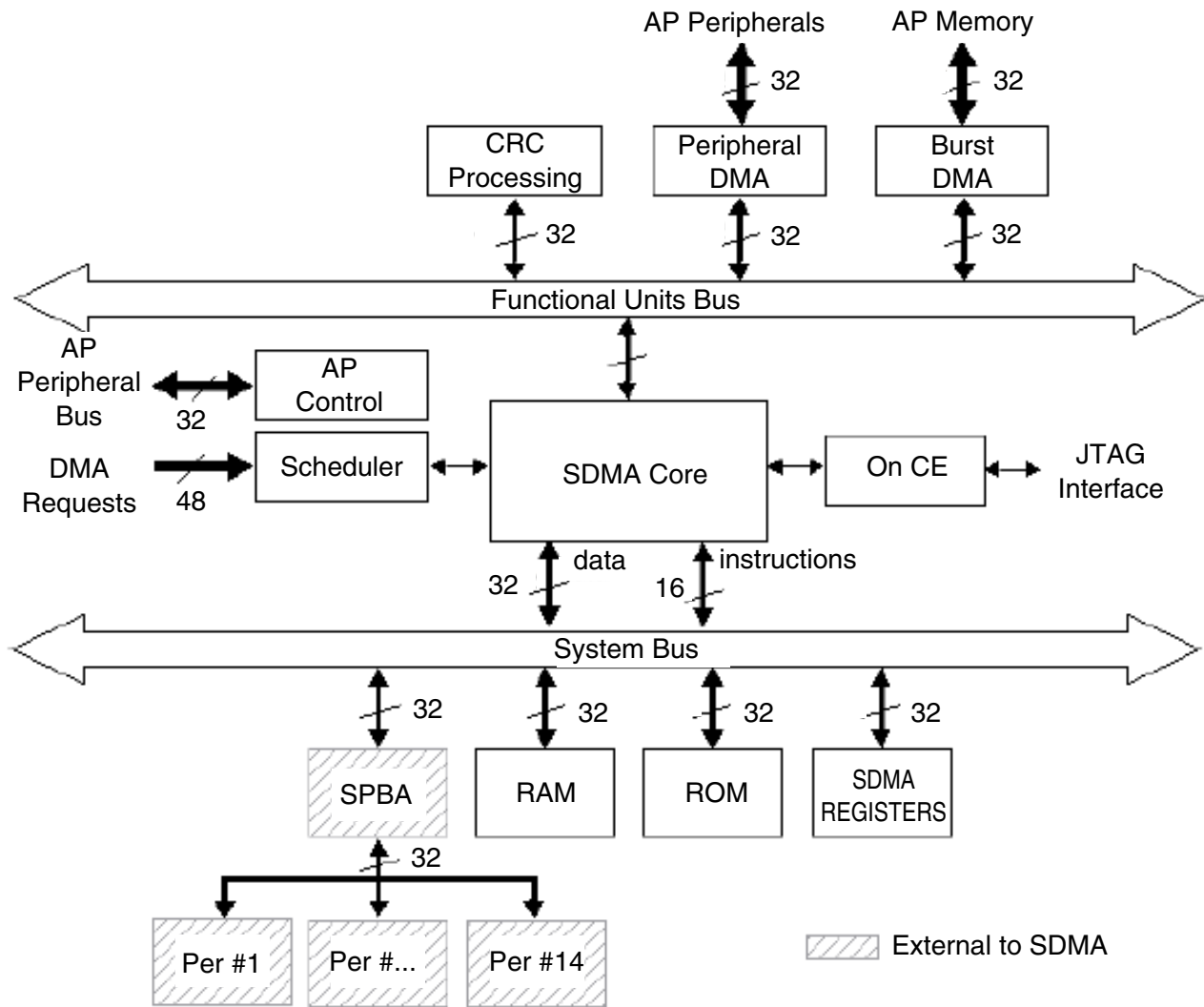
### 41.1 Overview

The Smart Direct Memory Access (SDMA) controller offers highly-competitive DMA features combined with software-based virtual-DMA flexibility. It enables data transfers between peripheral I/O devices and internal/external memories.

The SDMA controller helps maximize system performance by off-loading the ARM core in dynamic data routing.

#### 41.1.1 Block Diagram

The figure below shows a block diagram of the SDMA controller. It includes the custom RISC core along with its RAM, ROM, DMA units, and the scheduler.



**Figure 41-1. SDMA Block Diagram**

The SDMA core executes short routines that perform DMA transfers; these routines are called *scripts*. The SDMA core interfaces to its own memory via the SDMA system bus. The SDMA system bus supports a 32-bit data path and a 16-bit address bus. The system bus datapath is used for both 16-bit instruction (program) memory access and 32-bit data access. DMA units interface to the core via the Functional Unit Bus and use dedicated registers to perform DMA transfers.

The SDMA memory contains a ROM and a RAM. The ROM contains startup scripts (for example, boot code) and other common utilities, which are referenced by the scripts that reside in the RAM. The internal RAM is divided into a context area and a script area (more details about this mapping are available in [Instruction Memory Map](#) and [Data Memory Map](#)).

Every transfer channel requires one context area to keep the contents of all the core and unit registers while inactive. Channel scripts are downloaded into the internal RAM by the SDMA using a dedicated channel that is started during the boot sequence. Downloads are invoked using commands and pointers provided by the ARM platform. Every channel contains a corresponding channel script located in RAM and/or ROM that can be reconfigured independently as-needed. Channel scripts can be stored in an external memory and downloaded when needed. The SDMA can be configured with any mixture of scripts to enable an endless combination of supported services.

The scheduler monitors and detects DMA requests, mapping them to channels, and mapping individual channels to a pre-configured priority. At any given point, the scheduler presents the highest priority channel that requires service to the SDMA core. A special SDMA core instruction is used to "conditionally yield" the current channel being executed to an eligible channel that requires service. If (and only if) there is an eligible channel pending, will the current channel execution be preempted.

There are two yield instructions that differently determine the eligible channels: In the first version, eligible channels are pending channels with a strictly higher priority than the current channel priority. In the second version (yieldge), eligible channels are pending channels with a priority that is greater or equal to the current channel priority. The scheduler detects devices that need service through its 48 DMA request inputs. After a request is detected, the scheduler determines the channel(s) that is (are) triggered by this request and marks it (them) as pending in the "Channel Pending (EP)" register. The priorities of all the pending channels are continuously evaluated in order to update the highest pending priority. The channel pending flag is cleared by the channel script when the transfer has completed.

The ARM platform control block contains the control registers used to configure the 32 individual channels. There are 48 Channel Enable registers, and every register maps one DMA request to any desired combination of channels. The 32 Priority registers are used to assign a programmable 1-of-7 level priority to every possible channel. This block also contains all other control registers that the ARM platform can access.

The 48 DMA requests that are connected to the scheduler come from a variety of sources. The "receive register full" and "transmit register empty" signals found in the UART and USB ports are typical examples of DMA requests that can be connected to the SDMA. These requests can be used to trigger a specific SDMA channel, or several channels.

There is an OnCE compatible debug port for product development. The OnCE includes support for setting breakpoints, single-step and trace, and register dump capability. In addition, all memory locations are accessible from the debug port.

## 41.1.2 Features

The following are the SDMA features:

- Multi-channel DMA supporting up to 32 time-division multiplexed DMA channels
- Hardware or software driven triggers for each channel
- 48 hardware driven triggers that can be mapped to any channel.
- Memory accesses including linear addressing, FIFO addressing and 2D addressing
- Fast context-switching with two-level, priority-based preemptive multi-tasking
- 16-bit instruction-set micro-RISC engine (the SDMA core)
- Two DMA units with some or all the following features:
  - Auto-flush and prefetch capability
  - Flexible address management (increment, decrement, and no address changes on source and destination address)
  - Misaligned data-transfer support
  - Uni-directional and bi-directional flows (copy mode)
  - Up to eight-word buffers for configurable burst transfers
- Support of byte-swapping
- An available API and library of scripts
- Little-Endian and Big-Endian modes
- Hardware handshakes for low-power entry sequence
- Security support to lock contents of the SDMA script RAM.
- 4-Kbyte ROM containing startup scripts (for example, boot code) and other common utilities that can be referenced by RAM-located scripts
- 8-Kbyte RAM area is divided into a processor context area and a code space area used to store channel scripts that are downloaded from the system memory
- Debug support, including a OnCE port, real-time monitors, and embedded cross-trigger events
- Supported clock frequencies in process:
  - Configurable clock options for the SDMA core and the ARM platform DMA units
    - 1:2 ratio with maximum of SDMA core running at ARM platform Peripheral Bus speed and DMA running at max DMA frequency.
    - 1:1 ratio when both SDMA core and ARM platform DMA clocks are set to the ARM platform Peripheral Bus speed.
- Peripheral bus interface for configuration register programming by the ARM platform
- The SDMA RISC engine (arithmetic and logic operations), which is referred to as the "SDMA core."
- An internal peripheral bus connected to the Shared Peripherals Bus Interface (SPBA) that enables access to up to 14 shared peripherals. SDMA supports 32-bit accesses to word peripherals and 16-bit accesses to half-word peripherals.

- The peripheral DMA unit that is hooked-up to the ARM platform Crossbar Switch to service ARM peripherals
- The burst DMA unit is able to perform burst accesses to the external memory
- All the DMA units are 32-bit AHB masters. They are connected to different buses, thus allowing concurrent accesses.

## 41.2 External Signals

The table found here describes the external signals of SDMA.

**Table 41-1. SDMA External Signals**

Signal	Description	Pad	Mode	Direction
SDMA_EXT_EVENT0	Event 0 signal	ECSPI2_MISO	ALT1	I
SDMA_EXT_EVENT1	Event 1 signal	ECSPI2_MOSI	ALT1	I

## 41.3 Clocks

The table found here describes the clock sources for SDMA

. Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information. For functional information regarding module clocks, see [SDMA Clocks and Low Power Modes](#).

**Table 41-2. SDMA Clocks**

Clock name	Clock Root	Description
events_sync_clk (clk)	ahb_clk_root	ARM peripheral / events clock
ips_hostctrl_clk	ipg_clk_root	Host control clock
ap_ahb_clk	ahb_clk_root	ARM platform bus clock
core_clk	ipg_clk_root	Module / Core clock
tck	-	JTAG access clock

## 41.4 Functional Description

The figure below shows the SDMA topology, and is composed of the following components:

## Functional Description

- SDMA Core ([SDMA Core](#))
- SDMA Scheduler ([Scheduler](#))
- Functional Units:
  - Burst DMA ([Burst DMA Unit](#))
  - Peripheral DMA ([Peripheral DMA Unit](#))
- ARM platform Control for ARM control register access.
- Internal RAM and ROM Memory ([SDMA Programming Model](#))
- OnCE debug Port ([The OnCE Controller](#))

The functional unit bus provides access by the SDMA core to the DMA units. The system bus provides access to SDMA internal memory and also supports up to 14 peripherals.



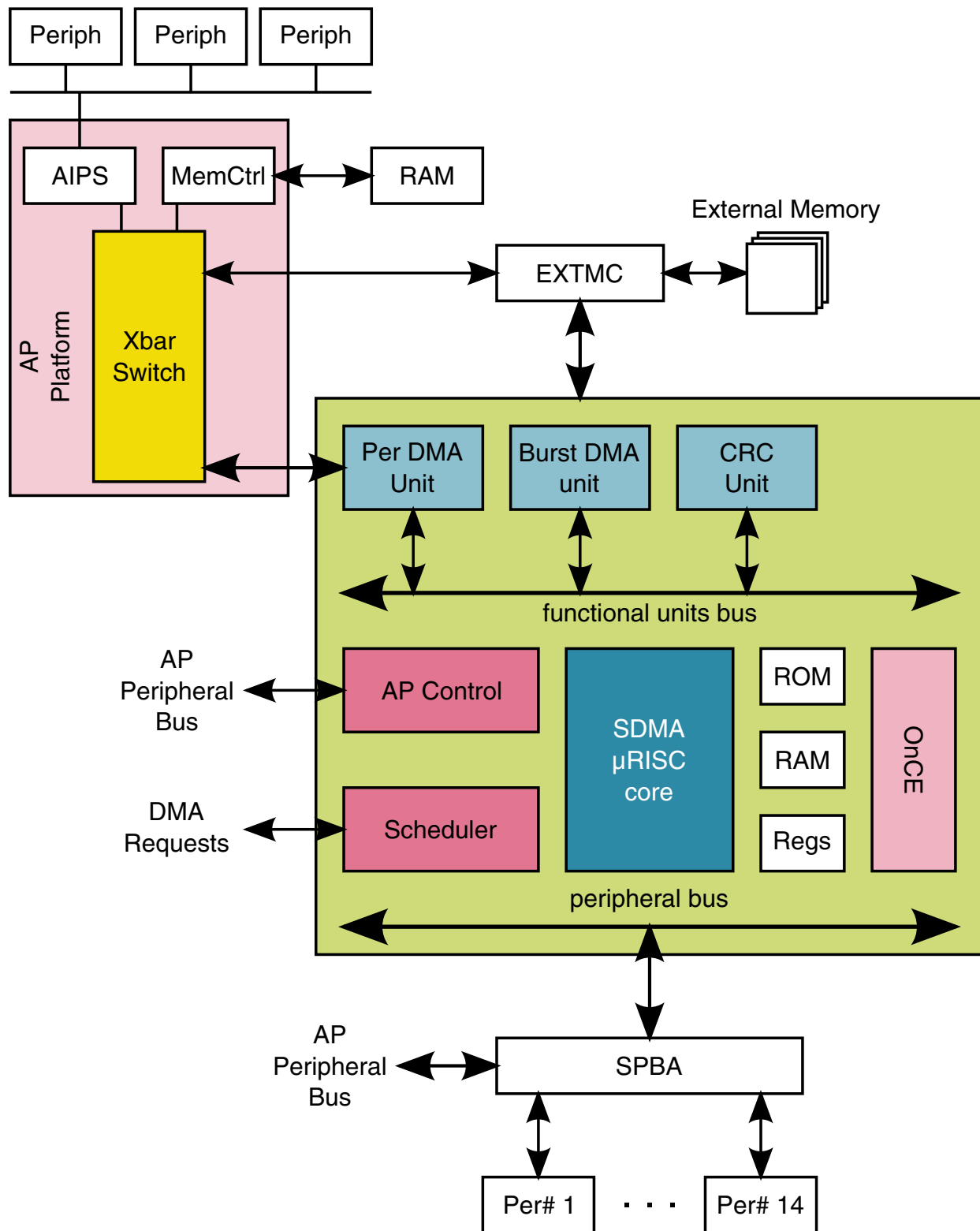


Figure 41-2. SDMA Connections

## 41.4.1 SDMA Core

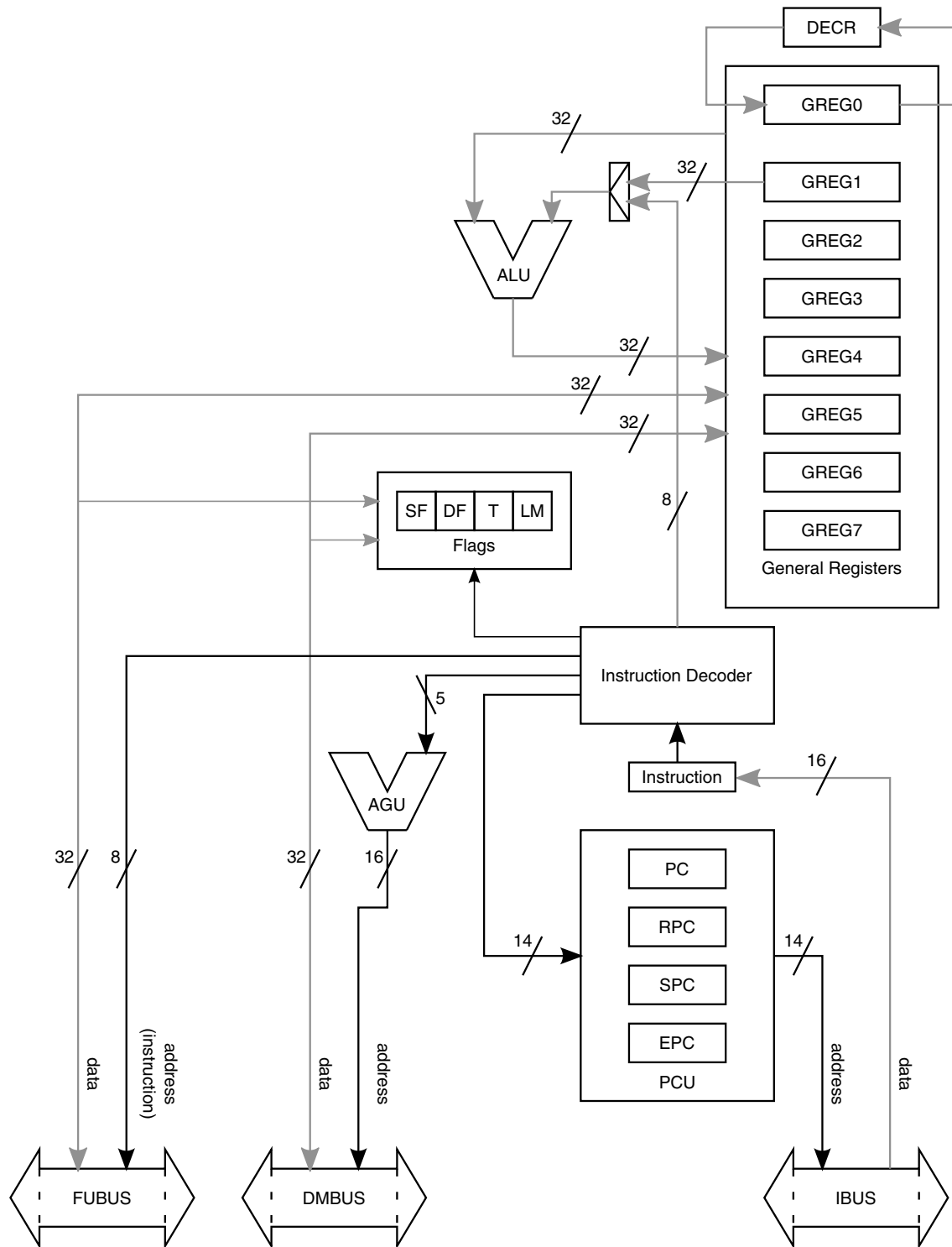
The SDMA core is a customized RISC-like processor that is specifically developed to control DMA units and perform L1 tasks like byte-stuffing or framing.

The SDMA core incorporates on-chip debug capability using the OnCE.

The SDMA core is based on a 32-bit register architecture with 16-bit instructions. There are eight general purpose 32-bit registers, four flags (T, LM, SF, and DF), and four PCU registers (PC, RPC, SPC, and EPC) that can address 16,384 16-bit instructions.

### 41.4.1.1 SDMA Core Structure

The figure found here shows the structure of the SDMA core. It also shows the different registers, calculation resources, and possible data movements.



**Figure 41-3. SDMA Core**

- The Program Control Unit (PCU) is described in [Program Control Unit \(PCU\)](#). It handles the state of the core and generates the instruction fetch addresses. Instructions are retrieved from the Instruction Bus (IBUS) and stored in the SDMA

core instruction register prior to their decoding. The PCU contains the following registers:

- The Program Counter (PC) contains the address of the current instruction.
- The Return Program Counter (RPC) contains the address of the instruction that follows a jump to the subroutine.
- The Start Program Counter (SPC) contains the address of the first instruction of the current hardware loop.
- End Program Counter (EPC) contains the address of the last instruction of the current hardware loop.
- The other core registers are the general purpose registers (GREGn) and the flags.
  - The general purpose registers can be used to hold data and addresses. They can be loaded with immediate values (for example, 8-bit data that are encoded in the instruction), results of calculations that were performed with the ALU, 32-bit data that comes from the memory or peripherals via the Data Memory Bus (DMBUS), 32-bit data that comes from the DMAs via the Functional Units Bus (FUBUS) or another general purpose register. Their content can be the operands of the ALU, the data to send on either bus (DMBUS or FUBUS), or a pointer to memory (DMBUS address).
  - The general register 0 (GREG0) is also the hardware loop counter. In hardware loops, it cannot be used for any other purpose. This register uses a dedicated decrement unit (DECR) shown in [Figure 41-3](#).
  - The flags reflect the status of operations:
    - SF and DF are set when the last load or store on either bus (FUBUS or DMBUS) received an error response.
    - LM is set when the core is executing instructions inside a hardware loop.
    - T is set when the ALU operation result was 0 or the loop counter reaches 0 (the latter is preponderant when an ALU operation is the last instruction of a hardware loop).
- The ALU has two operands: any general register and either a second general register or an immediate value. The result is always stored into the first general register. A NOP function can be utilized by moving a register's contents into itself (For example, the instruction: mov R0,R0).
- The 16-bit instructions are fetched via the instruction bus (IBUS) whose address is driven by the PC. The SDMA RAM and ROM are visible to the core as 16-bit devices through this interface.
- The memory (RAM and ROM), memory mapped registers, and external peripherals are accessed via the DMBUS. The address is always taken from a general register whose content is added to a 5-bit immediate value. This is the only available addressing mode. The DMBUS is a 32-bit data bus. Except for the peripherals that

are external to the SDMA, the address accuracy is the 32-bit word (for example, adding 1 to an address points to the next word, not the next byte).

- The functional units are accessed via the FUBUS connection. The data is exchanged with any general register, but the address (which in fact is the instruction and the selector of the functional unit) comes from an 8-bit field of the corresponding load or store.

### 41.4.1.2 Program Control Unit (PCU)

This part of the SDMA core is dedicated to the control of the RISC engine, as implied by the instructions that are executed. Its behavior is determined by the instruction type and the inputs of the SDMA.

It contains the PC, RPC, SPC, and EPC registers that are described in [SDMA Core Structure](#).

#### 41.4.1.2.1 Instruction Types

The state sequence and the delay of execution vary according to the type of the instruction. There are six possible categories of instructions, as follows:

1. Standard: Most of the instructions belong to this category, and always last 1 cycle.
2. ldf/stf: These are respectively the load and store instructions that access the functional units. They last  $1+n$  cycles where  $n$  is the number of wait-states of the targeted functional unit.
3. ld/st: These are the load and store instructions that access the memory and peripherals. They last  $1+n$  cycles where  $n$  is the number of wait-states of the targeted device (1 for the ROM, RAM, and memory mapped registers, 1 + the external peripheral wait-states). These instructions always last at least two cycles, but the core is able to handle them in one cycle. The first wait-state is inserted outside the core.
4. Branch: These are all the instructions that cause the Program Counter to point to another instruction other than the following one (for example, one that breaks the sequential flow). There are the absolute jumps, the conditional branches, the jump to the sub-routines, and the return from the sub-routine.
5. Loop, Modified Load or Store: The hardware loop instruction modifies the potential behavior of any load or store inside the loop (for example, when the LM flag is set). A jump may be implied after any such load or store if it received an error. The error causes an early exit of the loop, which means a jump to the instruction that follows the one that is pointed to by EPC. An additional cycle is required by the PCU to perform the jump (+1 to the ld/st/ldf/stf original execution delay). Although there is

usually an implicit jump after the last instruction of the loop when the PC goes back to SPC, this is performed at no cycle cost.

6. Done: The done, yield, or yieldge instructions are used to control channel switching. When no channel switching is performed, these instructions last a single cycle. When there is a change of channel or context switch, the delay is variable and depends on many factors (as detailed in [Context Switching](#)).

#### 41.4.1.2.2 PCU States

The PCU state is visible through outputs of the SDMA (see [Real-Time Debug Outputs](#)) or the OnCE status register(see [OnCE Status Register \(OSTAT\)](#)).

The PCU state is a four-bit field that can take the values shown in the following table. [Figure 41-4](#) shows the possible state transitions and the corresponding conditions.

**Table 41-3. PCU States**

Value	State	Description
0	Program	This is the usual instruction cycle.
1	Data	This state is inserted when there are wait-states during a load or a store on the data bus (ld/st type).
2	Change of Flow	This is the second cycle of any instruction that breaks the sequence of instructions (branch and done types). This state lasts only a single cycle; it is always followed by the Program state.
3	Error in Loop	This state is used when an error causes a hardware loop exit (loop-modified load or store type). This state only lasts a single cycle; it is always followed by the Program state.
4	Debug	The SDMA is stopped in debug mode.
5	Functional Unit	This state is inserted when there are wait-states during a load or a store on the functional units bus (ldf/stf type).
6	Sleep	No script is running: The core is idle after saving the last channel context.
7	Save	The context switch FSM is saving the current channel.
8	Program in Sleep	Same as Program except there is no associated channel, this state is used when instructions are executed after entering debug mode, whereas the core was in either Sleep mode.
9	Data in Sleep	This is the same as Data except there is no associated channel.
10	Change of Flow in Sleep	This is the same as Change of Flow except there is no associated channel. This state only lasts a single cycle, and is always followed by the Program in Sleep state.
11	Error in Loop in Sleep	This is the same as Error in Loop except there is no associated. channel. This state only lasts a single cycle, and is always followed by the Program in Sleep state.
12	Debug in Sleep	This is the same as Debug except the core was put in debug mode when no channel was active.
13	Functional Unit in Sleep	This is the same as Functional Unit except there is no associated channel.

*Table continues on the next page...*

**Table 41-3. PCU States (continued)**

Value	State	Description
14	Sleep after Reset	This shows that no script is running, and the core is idle after a reset. When a channel becomes active, no context is restored but the core starts its boot program located at address 0 (or the address available in register in <a href="#">Channel 0 Boot Address (SDMAARM_CHN0ADDR)</a> ).
15	Restore	The context switch FSM is restoring the next channel context.

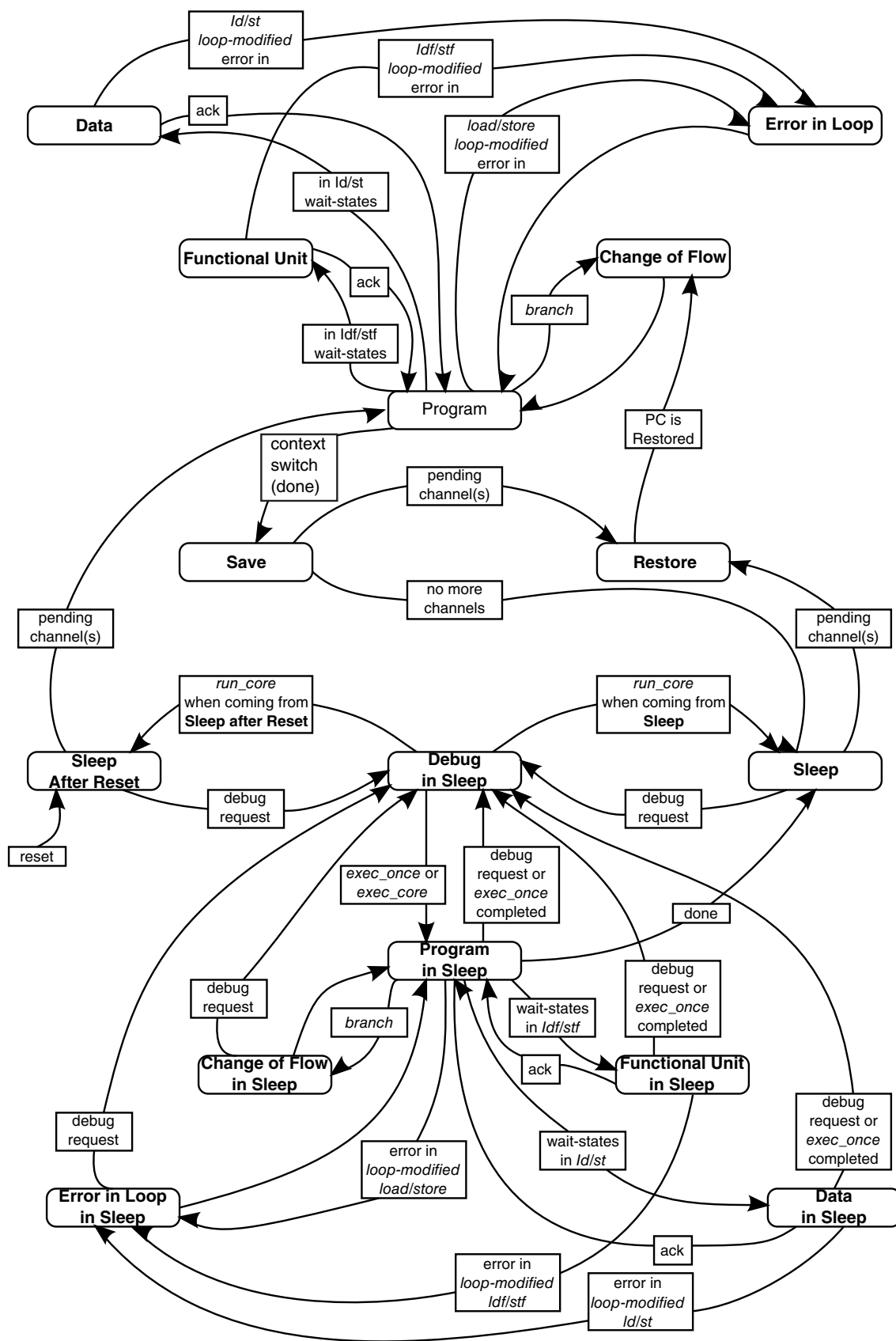


Figure 41-4. PCU State Diagram



### 41.4.1.3 SDMA Core Memory

The SDMA has two memory spaces: one for the instructions and one for the data. As both spaces share the same resources (ROM and RAM devices), the system bus manages possible conflicts when the core accesses the same resource for both an instruction read and a data read or write.

Program and data memory is further described in [Address Space](#).

Instructions of 16-bit width are stored in 32-bit wide devices and can be accessed as data. The mapping is Big Endian: an even instruction address (terminated by 0) accesses the most significant part of the 32-bit data (bits [31:16]), and an odd instruction address (terminated by 1) accesses the least significant part of the 32-bit data (bits [15:0]). Instructions can be fetched out of internal ROM or RAM.

Data can be read from ROM, RAM, memory mapped registers, and external peripherals, and written to the same devices (except the ROM).

The ROM contains bootload scripts, channel scripts, and common subroutines which may be referenced by channel scripts elsewhere in the ROM or RAM.

The RAM is divided into a context area and a code space area which may be used to store channel scripts. The RAM contains undefined values after a hardware reset. Channel scripts and initial context values are downloaded into RAM using channel 0 which is reserved for bootload functions.

## 41.4.2 Scheduler

All channel scheduling hardware is included in the Scheduler.

### 41.4.2.1 Primary Functions

The scheduler is a hardware-based design used to coordinate the timely execution of 32 virtual DMA channels by the SDMA core on the basis of channel status and priority.

The scheduler performs the following functions:

- Monitors, detects, and registers the occurrence of any one of the 48 DMA requests
- Links a specific request to a channel or group of channels (channel mapping)
- Ignores requests that are not mapped to a previously configured channel
- Maintains a list of all the channels that are requesting service

- Assigns a pre-programmed priority level (1 of 7) to every channel requesting service
- Detects and flags overrun/underrun conditions

### 41.4.2.2 Channels and DMA Requests

#### 41.4.2.2.1 Channels

A Virtual Channel (hereafter simply called a channel) manages a flow of data through the SDMA. Flows are typically unidirectional.

The SDMA can have up to 32 simultaneously operating channels, numbered from 0 to 31. Channel 0 is usually dedicated to control the SDMA script downloading. All the channels can be assigned by the ARM platform software.

#### 41.4.2.2.2 DMA Requests

A DMA request is caused by externally (for example, external to the SDMA) controlled conditions (for example, UART receive FIFO reaches a threshold). The SDMA currently supports up to 48 DMA requests.

#### 41.4.2.2.3 Mapping from DMA Requests to Channels and Priorities

A channel can stall waiting on a single DMA request. A single DMA request can awake more than one channel (in fact, any request can awake any combination of channels).

The mapping between DMA requests and channels is program-controlled. There is a storage element assigned for each of the 48 requests that contains a bitmap table of the channels that are awakened by the event.

Every channel also has a three-bit register that indicates its priority.

### 41.4.2.3 Scheduler Functional Description

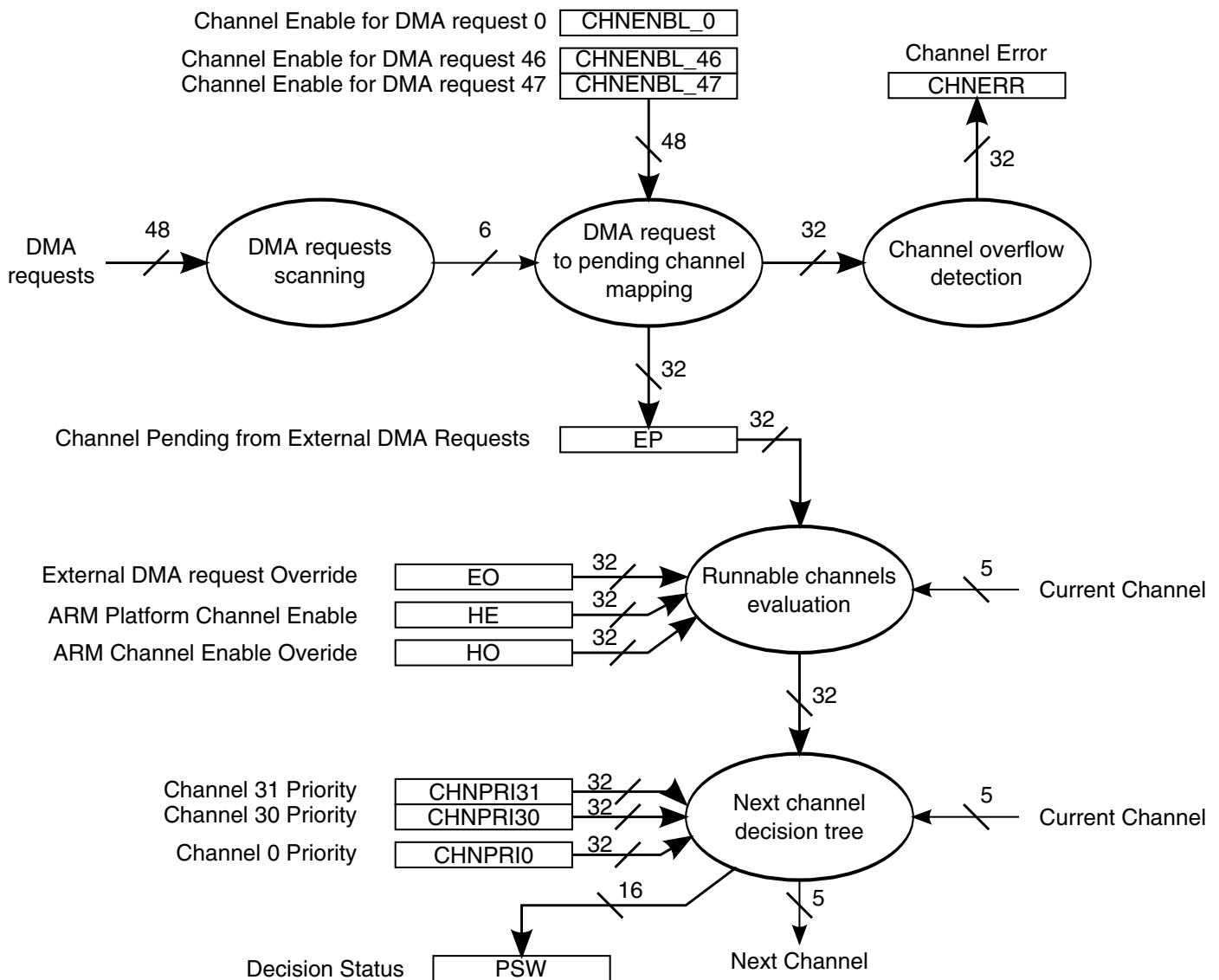
[Scheduler Overview](#) describes the behavior of the SDMA scheduler-from the channel enabling conditions to the highest priority pending channel selection.

### 41.4.2.3.1 Scheduler Overview

The scheduler algorithm is built in hardware. It is provided with possibilities for the ARM platform to control its behavior.

The scheduler processes incoming DMA requests, maps detected requests to 0, one, or several channels, maintains a list of channels that are requesting service (pending channels), identifies the top priority and its associated channel, and selects the next active channel when the current channel yields.

The following figure shows a functional overview.



**Figure 41-5. SDMA Hardware Scheduler**

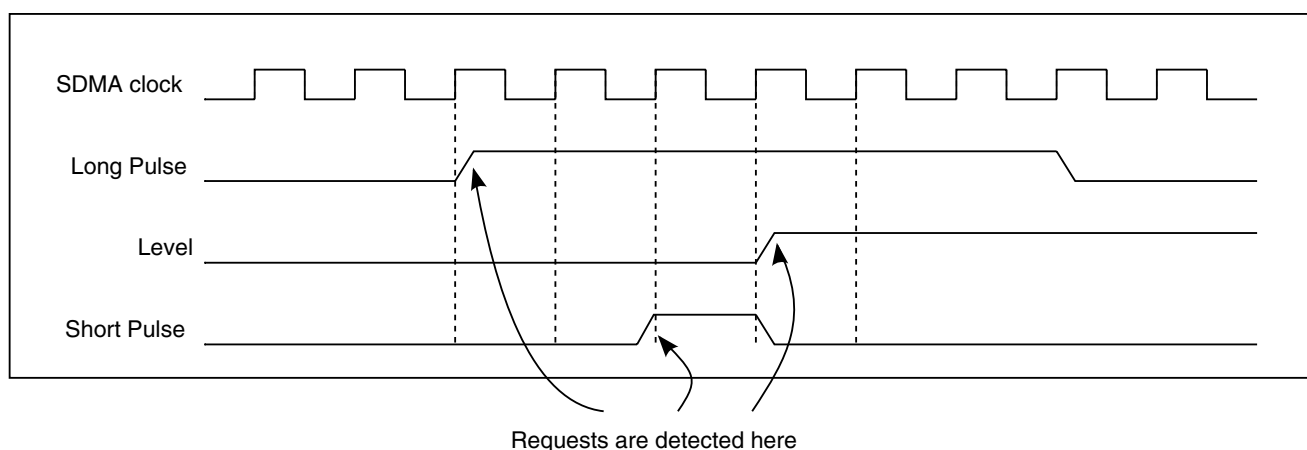
### 41.4.2.3.2 DMA Requests Scanning

The scheduler contains a 48-bit edge detection device that detects the rising edge of every DMA request and transmits the request number to the next stage.

The DMA requests are assumed to be generated on the same reference clock as the SDMA core clock; they are detected as soon as the signal goes from a 1-to-n-cycles low state to a 1-to-m-cycles high state.

This system is able to detect single-cycle pulses as well as level-based DMA requests such as a FIFO threshold crossing. In this case, the SDMA provides a memory mapped register that can be used by the channel script to monitor the DMA requests lines, and thus determines whether the data transfer is done or not done, and then continues with the transfer or closes the channel.

When several DMA requests are detected at the same time, they are forwarded to the next scheduler stage at the rate of one request per cycle. No request is lost.



**Figure 41-6. Examples of Valid DMA Requests**

The DMA request inputs are connected to various sources that depend on the SoC. The exact list of DMA request inputs and their associated number is available in each respective project-specific chapter.

### 41.4.2.3.3 Mapping DMA Requests to Pending Channels

Whenever a DMA request is detected by the first stage, its number is used in the second stage to determine the channels that have to be activated.

This is performed with an array of 48 registers that are 32 bits wide: There are 48 Channel Enable Registers (CHNENBLn), one register per DMA request. The DMA request number selects the Channel Enable Registers, and every bit of this 32-bit register indicates that the corresponding channel must be activated when it is a 1.

This information is passed on the EP register. For every bit of the Channel Enable Register that is set, the corresponding bit of the EP register is also set, and the remaining bits of EP are left unchanged. The transformation of EP is summarized by the following equation:

$$EP = EP \text{ or } CHNENBLn$$

The EP register is used to know which channels require service because they received a DMA request.

Typical contents of the CHNENBLn registers are all 0s, except for a single bit set. For example, a DMA request triggers one channel, but all 0s or several 1s are possible. One DMA request could activate several channels, and the channel execution sequence can be controlled by the channel priorities and numbers, as explained in the next sections. The following table illustrates an example configuration.

### NOTE

From the table, the DMA request 0 is programmed to simultaneously trigger channels 0, 1, and 31. Also, DMA requests 30-47 are not used in this example. The remaining channels 2 to 30, are configured to be triggered by DMA requests 29 to 1, respectively.

**Table 41-4. Channel Enable RAM Programming Example**

DMA Request Number	Channel																															
	31																															0
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1
1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
3	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
8	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Table continues on the next page...

**Table 41-4. Channel Enable RAM Programming Example (continued)**

[illegible]

*Table continues on the next page...*

**Table 41-4. Channel Enable RAM Programming Example (continued)**

DMA Request Number	Channel																															
	3 1																															0
44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
46	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### 41.4.2.3.4 Channel Overflow

A channel overflow occurs when a DMA request requires service from channel  $n$  by setting bit  $n$  of the register EP, but this bit is already set, meaning channel  $n$  is already pending. This can come from an overrun/underrun condition.

This detection is possible only when the DMA requests are pulses, because a level-based DMA request stays high until it is serviced, even though an underrun or overrun condition occurs, thus preventing another edge detection of the DMA request.

The channel overflow information is saved in the 32-bit CHNERR register (1 bit per channel). You can configure the SDMA to trigger an interrupt to the ARM platform when there are 1s in CHNERR. Every bit of CHNERR is masked with the corresponding bit of INTRMASK and if it gives a 1, the corresponding bit of INTR is set, triggering the interrupt.

#### 41.4.2.3.5 Runnable Channels Evaluation

The EP register is used in conjunction with several other 32-bit registers to determine the channels that are runnable.

Registers EO, DO, HO and HE, are controlled by the ARM platform. EP is controlled by the DMA requests and their mapping to channels.

Several channels may be runnable at any given time. The  $i^{\text{th}}$  channel is runnable if (and only if) the condition below is true:

$(\text{HE}[i] \text{ or } \text{HO}[i]) \text{ and } (\text{DO}[i]) \text{ and } (\text{EP}[i] \text{ or } \text{EO}[i])$

After reset, the HE[i], HO[i], EP[i], and EO[i] bits are all cleared whereas the DO[i] bits are all set. The functions associated with DO are not available for this device. When DO[i] is set, the scheduler condition becomes:

(HE[i] or HO[i]) and (EP[i] or EO[i])

The registers in these equations are controlled as follows:

- ARM platform (host) channel enable flag HE[i] may be set or cleared by the ARM platform with the HSTART and STOP\_STAT registers. It can also be cleared by the i<sup>th</sup> channel script.

Typical usage is for the ARM platform to set this flag to activate the channel. The flag is cleared by the SDMA core when the transfer is done.

- Externally triggered channel pending flag EP[i] is set by the scheduler when the channel was activated by a DMA request. It can be cleared by the i<sup>th</sup> channel script.
- The ARM platform channel override flag HO[i] may be set or cleared by the ARM platform. When set, it enables the i<sup>th</sup> channel to run without the involvement of the ARM platform.

Typical usage is for the ARM platform to set this flag for channels that do not need ARM platform supervision such as channels that are controlled by DMA request events (EP).

- DO should always be set to 1 so that the runnable channel evaluation considers only HO, HE, EP, and EO.
- Externally triggered channel override flag EO[i] may be set or cleared by the ARM platform. When set, it prevents the i<sup>th</sup> channel from stopping and stalling on incoming peripheral DMA requests. This is the case when the channel is not handling data transfers with peripherals (for example, a memory to memory transfer).

The SDMA can clear the HE[i], and EP[i] bits by means of a done or notify instruction. The done instruction causes a reschedule; thus, enabling another channel to preempt the current one, while the notify instruction does not. The done and notify instructions can clear either HE[i] or EP[i] (never more than one at a time).

**Table 41-5. Runnable Channel Selection Control**

Register	Set by	Cleared By
HO	Write to HOSTOVR register	Write to HOSTOVR register
HE	Write to HSTART register	Write to STOP_STAT register or by the channel script with the done or notify instructions.
DO	Write to DSPOVR register	Write to DSPOVR register
EO	Write to EVTOVER register	Write to EVTOVER register
EP	Set by external DMA request event input.	By the channel script with the done or notify instructions



#### 41.4.2.3.6 Next Channel Decision Tree

The next channel number is computed from the runnable channels list, the current channel number, and their respective priorities.

It is re-evaluated every cycle, but is only used when the current channel yields or terminates by executing a yield, yieldge, or done instruction.

The decision tree is based on the selection of the runnable channel that has the highest priority.

The highest priority channel is selected according to the following rules:

- Runnable channels are sorted by priority.
- If one of the channels with the highest priority had been preempted by a channel with a higher priority, but did not want to yield to a channel of the same priority (for example, it executed a yield, not a yieldge), it is elected as the next channel.
- The channels that belong to the highest priority group are sorted by their number and the channel that has the highest number in this group becomes the next channel. For example, if priorities are the same, channel 31 will be selected before channel 30.

When the current channel requires a reschedule with a yield(ge) or a done instruction, the context switch decision is based on the instruction parameter, the current channel number and priority, and the next channel number and priority. The possible cases are all listed in the following table. The grayed cells correspond to unusual cases that should not occur with a typical usage of the SDMA.

**Table 41-6. Channel Switching Decision with a yield, yield(ge), or done**

Instruction	Current Channel	Next Channel	Priorities Comparison	New Running Channel/Comments
yield (done 0)	Runnable	Not runnable	none	Current
	Runnable	Runnable	Current > Next	Current
			Current = Next	Current
			Current < Next	Next <sup>1</sup>
	Not runnable	Not runnable	none	none <sup>2</sup> (occurs when the channel was disabled by the ARM platform)
	Not runnable	Runnable	none	Next <sup>1</sup> (occurs when the channel was disabled by the ARM platform)

*Table continues on the next page...*

**Table 41-6. Channel Switching Decision with a yield, yield(ge), or done (continued)**

Instruction	Current Channel	Next Channel	Priorities Comparison	New Running Channel/Comments
yieldge (done 1)	Runnable	Not runnable	none	Current
	Runnable	Runnable	Current > Next	Current
			Current = Next	Next <sup>1</sup>
			Current < Next	Next <sup>1</sup>
	Not runnable	Not runnable	none	none <sup>2</sup> (occurs when the channel was disabled by the ARM platform)
	Not runnable	Runnable	none	Next <sup>1</sup> (occurs when the channel was disabled by the ARM platform)
done (done>1)	Not runnable	Not runnable	none	none <sup>2</sup>
	Runnable	Not runnable	none	Current <sup>3</sup> (occurs when the done instruction does not disable the channel runnable condition)
	Not runnable	Runnable	none	Next <sup>1</sup>
	Runnable	Runnable	none	Current <sup>3</sup> (occurs when the done instruction does not disable the channel runnable condition)

1. Current channel script execution is stopped, its context is saved; the next channel context is restored and its script execution resumes
2. Current channel context is saved and SDMA enters IDLE mode
3. Current channel context is saved, then restored, and the current channel script resumes execution

Finally, when the SDMA is in IDLE mode and a runnable channel is elected as the next channel, its context is immediately restored and the script execution resumes.

The *combinatorial-decision* tree supports dynamic modifications of the EP, EO, HE, HO, and DO flags as well as dynamic modifications of the channel priorities. The propagation times are detailed in [Scheduler Pipeline Timing Diagram](#).

The decision tree status is available in the PSW register, which is continuously updated. It contains the next channel priority, the next channel number, the current channel priority, and the current channel number. When a priority is read as 0, it means the channel is not runnable.

A few examples of decisions are presented below:

- Channel 31 is running with priority 5, channels 13 and 24 are pending with the same priority 5; channel 24 is eligible as the next channel since  $24 > 13$ .
- Channel 31 is running with priority 7, channels 13 and 24 are pending with priority 5; channel 31 is the next channel because its priority is greater than the other pending channels.

- Channels 7, 23, and 29 are pending with the same priority. Channel 7 is active and runs a yieldge; it is preempted by channel 29. After a period of time, channel 29 runs a yieldge, it is then preempted by channel 23 that is the selected channel since channel 29 is the current channel. Later, channel 23 runs a yieldge and is preempted by channel 29. Channels 23 and 29 will go on switching after every yieldge until one of them terminates. It is only at that point that channel 7 becomes eligible again.
- Channel 11 is running with priority 3, and channel 15 is pending with priority 4. When the channel 31 script executes a yield instruction, it gets preempted by channel 15; then channels 6 and 18 with priority 3 become pending. Because channel 11 was preempted after executing a yield and there is no pending channel with a strictly greater priority, it is eligible as the next channel (although its number  $11 < 18$ ).

#### 41.4.2.3.7 Scheduler State Diagram

The [Figure 41-7](#) summarizes the behavior of the SDMA scheduler with details about the exact mechanism of the priority decision tree. It is important to understand the scheduler is a hardwired pipeline, which means all the stages are performed simultaneously every cycle, but a change on any given stage is reflected on the next stage after the delays presented in [Scheduler Pipeline Timing Diagram](#).

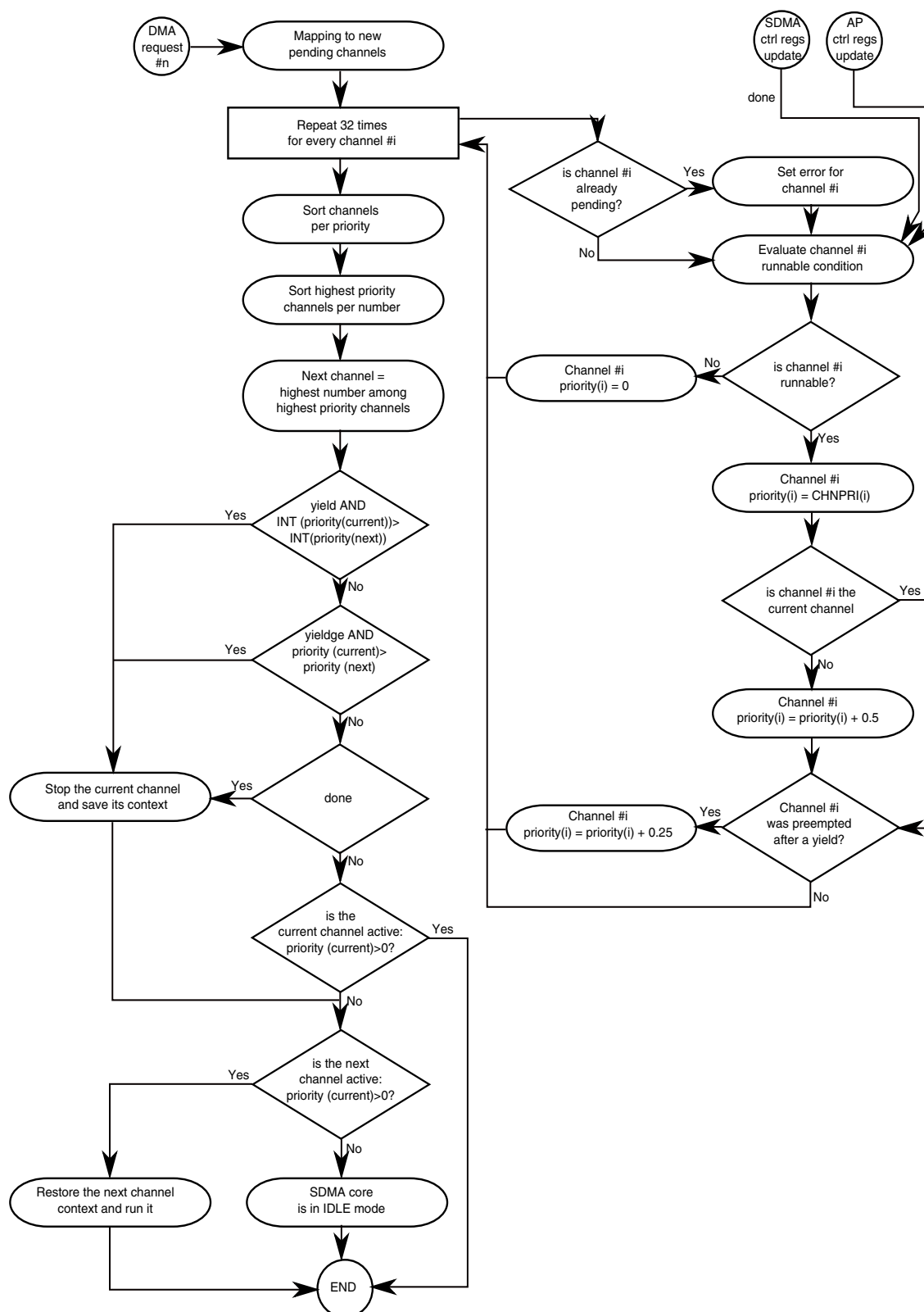
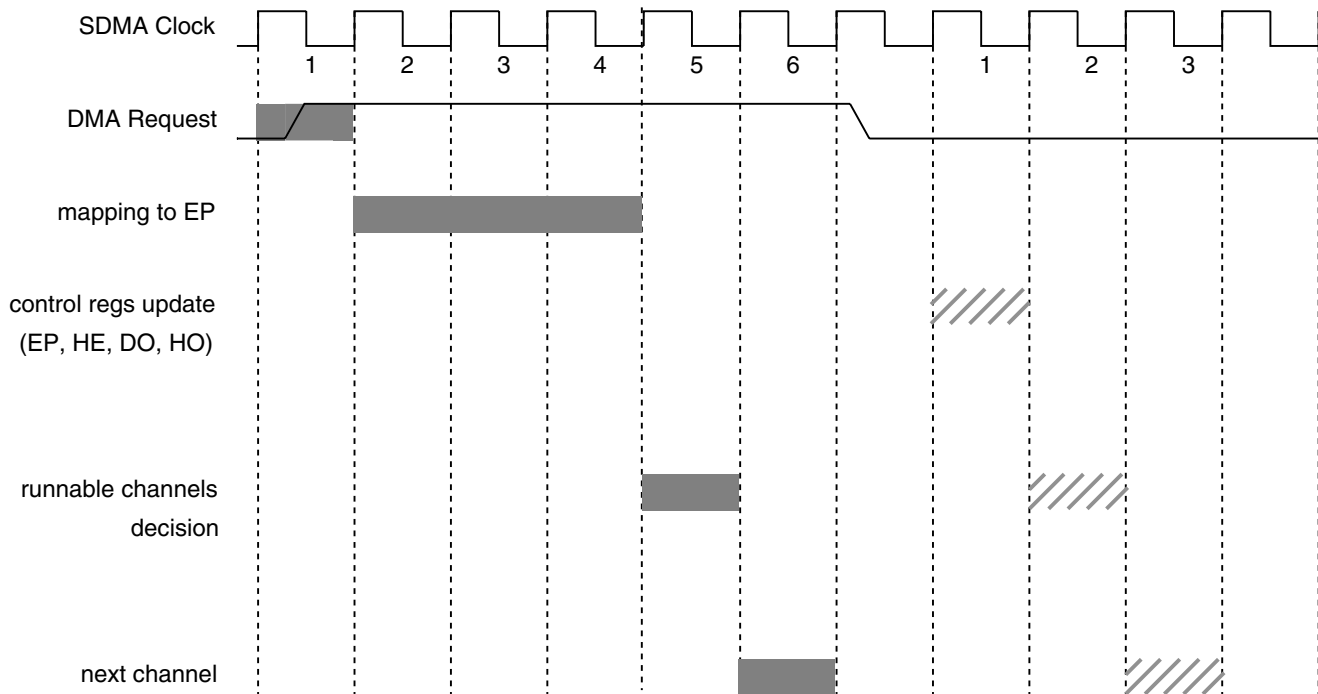


Figure 41-7. Scheduler State Diagram

#### 41.4.2.3.8 Scheduler Pipeline Timing Diagram

The SDMA scheduler process of DMA-request and control-register modifications is not immediate.

The figure below shows the exact delays of all the tasks. The reference clock is the SDMA core clock.



**Figure 41-8. Scheduler Timing Diagram**

Two numbers can be inferred from this timing diagram. First, it takes six SDMA core clock cycles to update the next channel from a DMA request. Second, it takes three SDMA core clock cycles to update the next channel from a direct modification of the condition registers (EP, DO, HE, or HO) by any processor. The processors that can modify these bits include SDMA with a done instruction or the ARM platform with a write access through the corresponding control port on their respective peripheral bus).

#### 41.4.2.3.9 Channel-DMA Request Mapping

The 48 DMA request inputs to the SDMA scheduler are listed in project-specific chapters. Refer to the respective chapters for this information.

#### 41.4.2.3.10 Examples: How to Start a Channel

A channel can be started when the following equation is true for channel  $i$ :

$(HE[i] \text{ or } HO[i]) \text{ and } (DO[i]) \text{ and } (EP[i] \text{ or } EO[i])$

Once this equation is true, the scheduler can start this channel according to the priority of all pending channels. Several examples of configuration are listed below:

1. To start a channel triggered by ARM platform software:
  - Initially, configure  $HO[i]=0$ ,  $DO[i]=1$ , and  $EO[i]=1$  using registers indicated in [Table 41-5](#).
  - ARM platform software triggers the channel by writing to the HSTART register to set  $HE[i]=1$ , thereby setting the above equation true.
2. To start a channel triggered by DMA request event.
  - Initially, configure  $HO[i]=1$ ,  $DO[i]=1$ , and  $EO[i]=0$  using registers indicated in [Table 41-5](#).
  - The DMA request is asserted to trigger the channel by setting  $EP[i]=1$ , which makes the above equation true.

#### 41.4.2.4 Context Switching

On execution of a done or yield(ge) instruction, the current channel may be changed either because it has finished (which necessarily happens when the done instruction is executed), or it was preempted by a higher priority channel (which is possible but not systematic when the yield(ge) is executed).

Upon a channel change the SDMA goes through a context switch procedure.

When the current channel yields or ends, the context for that channel is saved into the context RAM locations for that channel. When the next channel starts running, its context is first restored from RAM.

Since context RAM is not yet initialized by reset, there will be no context restore at the beginning of the first channel (bootload channel) run after reset. It is expected that the bootload channel will be used to initialize the context for all other channels. When the bootload channel finishes running or yields, SDMA will enter its SAVE state and save that channel's context into RAM. Then, if the bootload channel is called again later, the context will be restored from RAM when the channel starts again.

The context structure for each channel is defined in [Context Switching-Programming](#) and [Table 41-11](#). There will be one context area reserved for each channel. When a channel ends or yields, the SDMA core registers are automatically saved into the context RAM and later restored from the context RAM when the channel is next run. The total RAM space reserved for 32-channel contexts is either 3K or 4K depending on whether the SMSZ bit is set in the CHN0ADDR register, which enables an additional 8 words of scratch RAM for each context.

#### 41.4.2.4.1 Context Switch Modes

The exact procedure to save the context of the old channel, and to restore the context of the new channel depends on the context switch mode selected by the ARM platform in the CONFIG control register.

The following are the context switch modes:

- By default, the "dynamic" context switch is set. This mode provides the most efficient context switch for an average of eight cycles to stop the current channel, save its context, restore the next channel context, and resume its execution. It consists of saving modified registers of the current channel in the background (for example, during the channel execution)-which leaves very few registers to save when the switch is decided-resuming execution of the next channel as soon as possible (for example, when the minimal set of registers is restored), and continuing the restore phase during this execution.
- In "dynamic with no loop" mode, the same principle is followed except the modified registers are only saved in the background when the loop flag is not set. This mode offers almost the same effectiveness as the previous one, but it prevents the system from accessing the RAM during loops to save power. This is the recommended mode for an efficient context-switch when the loop bodies are short.
- In "dynamic power" mode, no background saving is performed, which reduces power consumption to the minimum. The modified registers are only saved when the context switch starts. The restore phase is the same as before. This is the mode that achieves the optimal power consumption at the cost of a slower context-switch.
- In a "static" context switch, all the registers are saved when a context switch is decided, and all the registers are restored before starting the execution of the new channel. This mode enables a predictable behavior of the context switch since all the registers are restored prior to the channel start and all registers are saved after the channel termination.

#### NOTE

Static context mode should be used for the first channel called after reset to ensure that the all context RAM for that channel is initialized during the context SAVE phase when the channel is done or yields. Subsequent calls to the same channel or different channels may use any of the dynamic context modes. This will ensure that all context locations for the bootloader channel are initialized, and prevent undefined values in context RAM from being loaded during the context restore if the channel is re-started later.

#### 41.4.2.4.2 Context Switch Procedure

The Program Control Unit goes into the *save* state, the current context is spilled into memory, and the next channel context is restored according to the context-switch mode that was selected by the ARM platform.

The context switch procedure is as follows:

1. Load the current context's spill base address.
2. Spill the modified registers of the current channel to memory according to the selected context switch mode while the channel is running.

On a done or yield(ge) that causes the channel preemption, the PCU goes into the *save* state. In *static* mode, all the registers are saved; whereas, in either *dynamic* mode, the registers that were modified but not yet saved are then saved, and the PCU registers and flags are finally saved.

3. Put the SDMA core into *sleep* and wait for new channels to be serviced. This step is skipped if there are pending channels when the current channel is saved.

As soon as there is at least one pending channel, the PCU goes into its *restore* state to restore the context of the channel that was elected by the scheduler.

Once a channel is elected, it remains the current channel until its script requests a rescheduling operation with a done or yield(ge) instruction. That means the current channel cannot be modified by the ARM platform, even if it is no more runnable or if its priority is modified.

The ARM platform can however force a reschedule by writing the corresponding bit in the CONFIG register, which has the same effect as if the script had executed a done instruction. That feature should only be used to stop the SDMA in emergency cases.

4. Load the context base-address of the new channel.

In "static" mode, all the registers are restored. In either "dynamic" modes, only the PCU registers are restored.

The new channel is running. In "static" mode, no more activity regarding context restoring or saving is performed. In either "dynamic" modes, the registers are restored in the background every time an access to the context RAM is possible, and priority is given to restoring the registers that are required by the next instruction to be executed. When a register has not been restored and the next instruction needs it, this instruction gets stalled until the register was restored.



In "dynamic" and "dynamic with no loop" modes, background saving of dirty registers is performed every time an access to the context RAM is possible and allowed by the context switch mode.

### NOTE

The contents of a channel context space in the context RAM depends on the selected context switch mode. In "dynamic" and "dynamic with no loop" modes, the contents of the context RAM tend to match the contents of the SDMA registers (except for the PCU registers and flags that are never saved in the background). In "dynamic power" and "static" modes, the contents of the context RAM remain unchanged until the channel terminates with a done or gets preempted.

#### 41.4.2.4.3 Context Map in Memory

Refer to [Context Switching-Programming](#).

### 41.4.3 Functional Units

The functional units are small systems that are used by the SDMA core to handle data transfers between the core and a bus domain external to the SDMA.

The SDMA core is able to control and exchange data with these systems by sending instructions and reading or writing data from/to the functional units' registers via the FUBUS. This is done with the ldf and stf instructions.

The following sections provide introductions to the available functional units. [Functional Units Programming Model](#) provides descriptions the functional units' behaviors.

#### 41.4.3.1 Burst DMA Unit

The burst DMA unit enables the SDMA core to perform data transfers to and from the ARM platform memory.

It is optimized for accessing SDRAM-like devices. It does not provide control to assign a privilege level to the DMA access. The burst DMA unit provides the SDMA with means to do the following:

- Perform up to 8-beat read and write bursts to the ARM platform memory, which optimizes throughput when accessing SDRAM-type devices because of an internal, 36-byte FIFO
- Access the ARM platform memory at once or twice the SDMA core frequency
- Copy data from one ARM platform memory location to another ARM platform memory location at the ARM platform bus speed, which provides a very high throughput
- Control the method for addressing the ARM platform memory (automatic increment of addresses or frozen addresses-the former aimed at accessing RAM-like memory and the latter aimed at accessing single-address FIFOs)
- Enable or disable automatic prefetch when reading data from the ARM platform memory. When the prefetch mode is selected, the burst DMA automatically triggers external bursts to fill its FIFO without waiting for the SDMA core to request the corresponding data, greatly improving throughput.
- Rely on the DMA to automatically flush its FIFO content when there is enough data to generate an 8-beat burst to the ARM platform memory. Or, it forces a flush when a data transfer must terminate.
- In the former case, the SDMA core may only be stalled when it tries writing data and there is not enough room left in the FIFO. In the latter case, the core is stalled until the data is effectively written to the ARM platform memory.

In automatic flush mode, the core receives an acknowledge that does not reflect the actual error status when the data is effectively written into the ARM platform memory. This error status is retrieved by a later access to the burst DMA.

Terminating a write data transfer with a forced flush command guarantees that any bus error to the ARM platform memory is caught.

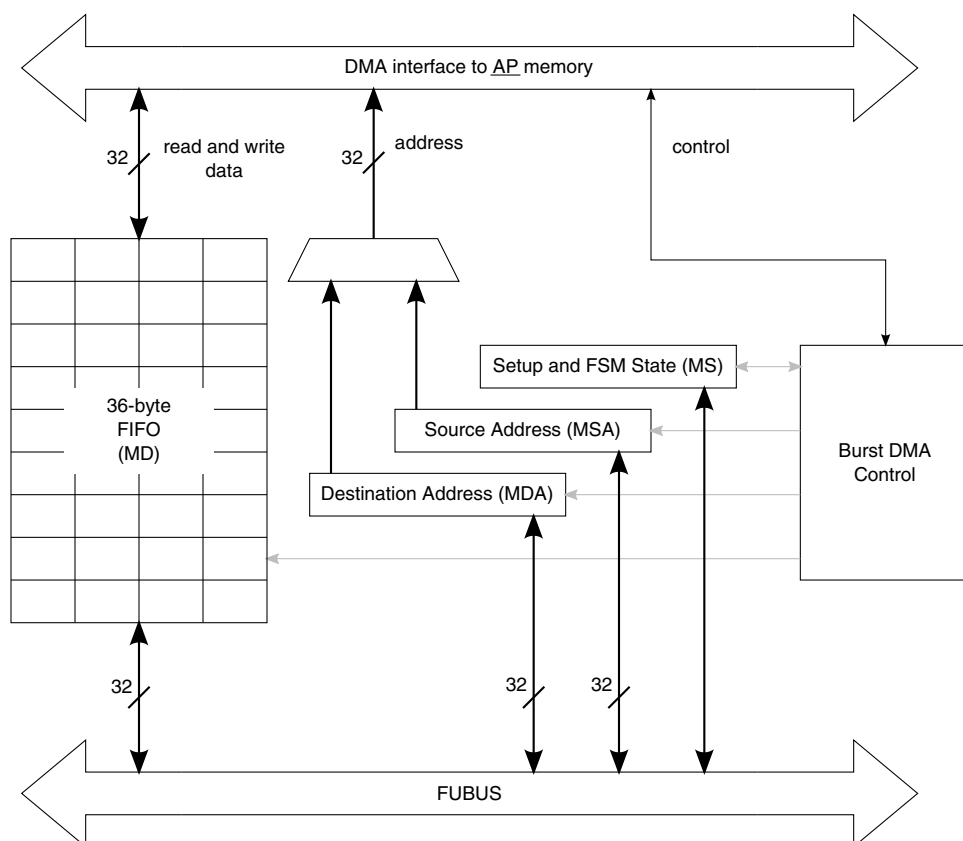
- Handle address alignment issues between the ARM platform memory map and the SDMA core data. This enables the core to read or write 32-bit data from the burst DMA, whereas the corresponding ARM platform address is not 32-bit aligned. This drastically improves the SDMA scripts' efficiency since the same loop that transfers 32 bits at a time can be used regardless of the start and end addresses in the ARM platform memory space.

This unit structure and registers are described in [Burst DMA Structure](#) and [Burst DMA Registers](#).

### 41.4.3.1.1 Burst DMA Structure

The burst DMA is essentially made up of a 36-byte FIFO, address registers, and a controlling state-machine. The 36-byte FIFO enables eight-word buffering with address alignment, and the state-machine manages clock adaptation when required.

The burst DMA is depicted in the figure below.



**Figure 41-9. Burst DMA Structure**

### 41.4.3.1.2 Burst DMA Registers

There are four registers, as follows, that may be accessed from the SDMA core:

- **MSA (Memory Source Address)** - Holds the source byte address in the ARM platform memory map for reading data from this location. This register is automatically modified every time the core reads new data from the FIFO.
- **MDA (Memory Destination Address)** - Holds the destination byte address in the ARM platform memory map for writing data to this location. This register is automatically modified every time the core writes new data into the FIFO.
- **MD (Memory Data)** - Labels the 36-byte FIFO access point: Reading a byte, halfword, or word from MD respectively retrieves the first 1, 2, or 4 bytes of the

FIFO (for example, the bytes that were stored first by the DMA state-machine when transferring data from the ARM platform memory).

- When the FIFO does not hold as many bytes as required by the SDMA core, the core is stalled until the missing bytes are read from the ARM platform memory. In the case of prefetch mode, the DMA controller decides when it should start a burst to ARM platform memory in order to reduce the risk to not have the required data for the future accesses of the core. When there is no prefetching, a burst is triggered when the required data is not available in the FIFO.

Writing a byte, halfword, or word to MD stores 1, 2, or 4 bytes, respectively, at the end of the FIFO (for example, these bytes are transmitted to the ARM platform memory after all the other bytes that were previously stored in the FIFO). When the FIFO does not have enough room left to hold the written data, the SDMA core is stalled until a sufficient amount of FIFO contents are flushed out to the ARM platform memory. Flushing is decided by the DMA controller when there are enough bytes in the FIFO to perform the largest allowed burst to ARM platform memory (the exact size depends on the burst start address and the AHB 1 Kbyte boundary rule). However, the SDMA core has the ability to force the flushing operation at any time, for example, when at the end of the data transfer, prior to channel closure.

- MS (Memory Setup) - Contains the state of the burst DMA control, the two flags that define whether each address register is incremented after every access to the external memory, and another flag that is set when a bus error occurred.

#### 41.4.3.1.3 Burst DMA Data Transfers

Three typical usages have been identified that involve the burst DMA: the data transfer startpoint, the endpoint, or both.

Every case requires a different procedure, as listed in the following sections:

##### 41.4.3.1.3.1 Data Retrieval from the ARM platform Memory

The following steps retrieve data from ARM platform memory using the burst DMA unit:

- Set up the MS flags to reflect the mode for the source address (incremented or frozen according to the type of accessed device: memory or peripheral FIFO), then initialize the source address register itself (MSA).
- Read data from the FIFO using the *ldf MD* instruction as many times as needed. If an error occurred during the fetch from ARM platform memory, the DMA control tags the error status on the data and the SDMA core SF flag is set when reading this data from the FIFO.

#### 41.4.3.1.3.2 Storing Data Into the ARM platform Memory

The following steps store data from ARM platform memory using the burst DMA unit:

- Set up the MS flags to reflect the mode for the destination address (incremented or frozen according to the type of accessed device: memory or peripheral FIFO), then initialize the destination address register itself (MDA).
- Store data into the FIFO using the *stf MD* instruction as many times as needed.
- When the transfer is finished and if the DMA worked in automatic flush mode, force the flush of the FIFO. This instruction is stalled until all the FIFO data is effectively sent to the ARM platform memory and the error status of the transfer is available in the DF flag.

#### 41.4.3.1.3.3 Transferring Data Between Two ARM platform Memory Locations-Burst DMA Unit

The following steps copy data between two ARM platform memory locations using the burst DMA unit:

- Set up the MS flags to reflect the modes for the source and destination addresses (all the combinations are possible), then initialize the source address register (MSA) and the destination address register (MDA). Both addresses must be word-aligned.
- Use as many *stf MD* instructions with the *COPY* flag as needed. Every instruction triggers a burst read of a given number of words from the source address (this number is provided to the burst DMA via the SDMA core general purpose register, which is referenced in the *stf* instruction). Once all the data is loaded into the FIFO, the DMA empties it with a write burst of the same count to the destination address. The DMA acknowledges prior to instruction completion, which frees the SDMA core for other tasks at no delay cost.
- Once the transfer is done, there should be a final access to the burst DMA to check the error status.

### 41.4.3.2 Peripheral DMA Unit

The peripheral DMA unit is the second functional unit that connects the SDMA to the ARM platform memory.

Unlike the burst DMA, it does not support burst transfers and is optimized for accessing peripherals. It does not provide control to assign a privilege level to the DMA access. Its feature list comprises the following:

- Access to the ARM platform peripherals or memory at once or twice the SDMA core frequency

- Data copy from one ARM platform memory location to another ARM platform memory location at memory bus speed, improving throughput
- Control of the method for addressing the ARM platform memory (automatic increment or decrement of addresses or frozen addresses, the first ones aimed at accessing RAM-like memory and the last one aimed at accessing single-address FIFOs)
- Selectable automatic prefetch when reading data from the ARM platform memory. In prefetch mode, the peripheral DMA automatically fetches another data-without waiting for the SDMA core to request it-when its data register is empty, which improves the throughput
- Selectable automatic flush. In this mode, the SDMA core may only be stalled when it tries writing data and the previous write operation is not finished yet; whereas, in forced flush mode, the core is stalled until the data is effectively written to the ARM platform memory.
- In automatic flush mode, the core receives an acknowledge that does not reflect the actual error status when the data is effectively written into the ARM platform memory or the peripheral. This error status is retrieved by a later access to the peripheral DMA. Terminating a write data transfer with a forced flush command guarantees that any bus error to the ARM platform memory has been caught.

This unit structure and registers are described in [Peripheral DMA Structure](#) and [Peripheral DMA Registers](#).

### 41.4.3.2.1 Peripheral DMA Structure

The peripheral DMA is made up of a 32-bit data register, two address registers, and a controlling state-machine. The state-machine manages clock adaptation, when required.

It is shown in the following figure.

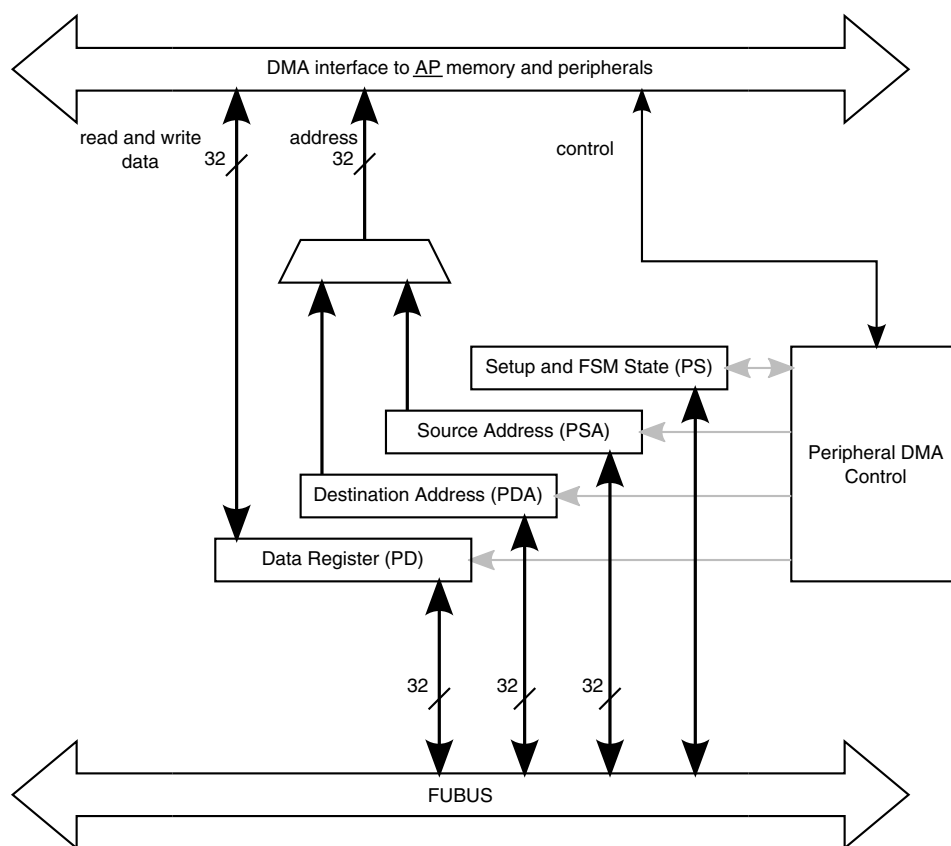


Figure 41-10. Peripheral DMA structure

### 41.4.3.2.2 Peripheral DMA Registers

According to [Figure 41-10](#), the peripheral DMA has four registers that may be read or written by the SDMA core:

- *PD (Peripheral Data)* is the DMA 32-bit data register.
- *PSA (Peripheral Source Address)* holds the source byte address in the ARM platform memory map for reading data from this location. This register is automatically modified every time the core reads a new data from PD.

- *PDA (Peripheral Destination Address)* holds the destination byte address in the ARM platform memory map for writing data to this location. This register is automatically modified every time the core writes a new data into PD.
- *PS (Peripheral Setup)* contains the state of the peripheral DMA control, two configuration fields that define the way address registers are modified after every data access, two additional configuration fields that define the data size to access the source and destination devices, and another field that contains the latest transfer error status.

### **41.4.3.2.3 Peripheral DMA Data Transfers**

There are three typical usages that involve the peripheral DMA, whether it is the data transfer start-point, endpoint, or both.

Every case requires a different procedure, as described in [Data Retrieval from the ARM platform Memory or Peripheral](#), [Storing Data into the ARM platform Memory or Peripheral](#), and [Transferring Data Between Two ARM platform Memory Locations-Peripheral DMA Unit](#).

#### **41.4.3.2.3.1 Data Retrieval from the ARM platform Memory or Peripheral**

The following steps retrieve data from ARM platform memory using the peripheral DMA unit:

- Set up the PS fields to reflect the mode and data size for the source (incremented, decremented, or frozen address register; 8-bit, 16-bit, or 32-bit data transfers), then initialize the source address register itself (PSA) with an address that is aligned to the programmed data size.
- Read data from PD using the ldf PD instruction as many times as needed. If an error occurs during the fetch from the ARM platform memory or peripheral, the DMA control tags the error status on the data and the SDMA core SF flag is set when reading this data from PD.

#### **41.4.3.2.3.2 Storing Data into the ARM platform Memory or Peripheral**

The following steps store data to ARM platform memory using the peripheral DMA unit:

- Set up the PS fields to reflect the mode and data size for the destination (incremented, decremented, or frozen address register; 8-bit, 16-bit, or 32-bit data transfers), then initialize the destination address register itself (PDA) with an address that is aligned to the programmed data size.



- Store data into PD using the *stf PD* instruction as many times as needed.
- When the transfer is finished and if the peripheral DMA worked in automatic flush mode, force the flush of PD. This instruction is stalled until PD contents are effectively sent to the ARM platform memory or peripheral, and the error status of the transfer is available in the DF flag.

#### 41.4.3.2.3.3 Transferring Data Between Two ARM platform Memory Locations-Peripheral DMA Unit

The following steps copy data between two ARM platform memory locations using the peripheral DMA unit:

- Set up the PS fields to reflect the modes and data size for the source and destination addresses (all the combinations of addressing modes are possible, but both data sizes must be identical), then initialize the source address register (PSA) and the destination address register (PDA). Both addresses must be aligned with the programmed data size.
- Use as many *stf PD* instructions with the *COPY* flag as needed. Every instruction triggers a single read from the source address; a single write of the received data immediately follows. The DMA acknowledges prior to instruction completion, which frees the SDMA core for other tasks at no delay cost.
- Once the transfer is done, there should be a final access to the peripheral DMA to check the error status.

### 41.4.4 SDMA Security Support

The SDMA provides support to SDMA software to block unauthorized updates to the scripts in RAM.

SDMA supports the following Security modes:

- Open Mode: has full control to load scripts and context into SDMA RAM. This is the default mode.
- Locked Mode: The ARM platform loads scripts and channel contexts at startup when it is still executing known safe software. When finished, it locks the SDMA to prevent further updates to RAM and selected registers. More details described in [Locked Mode](#).

#### 41.4.4.1 Locked Mode

The LOCK bit in the SDMA\_LOCK register provides support for SDMA scripts to freeze RAM contents after the initial bootload routine to prevent future unauthorized updates to SDMA RAM.

After initial RAM contents are uploaded, ARM platform software can set the LOCK bit to secure the RAM contents to prevent future updates by an unauthorized. After the LOCK bit is written with a '1', the SDMA is "locked" until reset.

The LOCK bit can be read in the SDMA's internal memory map in the LOCK register (see Section [SDMA LOCK \(SDMAARM\\_SDMA\\_LOCK\)](#)). SDMA scripts which load information into RAM can check the value of the LOCK bit to determine if an upload to RAM is allowed. If not allowed, the script can refuse to allow the request to copy data into the RAM to continue. The exact use of the LOCK bit in SDMA scripts for security control will be described in SDMA software documentation (see [SDMA Scripts](#)).

While SDMA is locked, attempts to write to the SDMA\_LOCK, CHN0ADR, ILLINSTADDR, and ONCE\_ENB registers will be ignored. All registers remain readable. Writes to other registers are still allowed.

Once the SDMA is locked, the LOCK bit can only be cleared by a reset. A hardware reset will always clear the LOCK bit. A software reset initiated by writing to the RESET register will only clear the LOCK bit if the SRESET\_LOCK\_CLR bit in the SDMA\_LOCK register is set. Since SDMA\_LOCK register cannot be updated if SDMA is locked, the SRESET\_LOCK\_CLR bit must be configured before setting the LOCK bit. The SRESET\_LOCK\_CLR bit will also be cleared by resets that clear the LOCK bit.

The SDMA RISC core uses the ILLINST and CHN0ADDR registers as pointers to determine where to jump to after an illegal instruction or upon boot after a reset. The LOCK bit prevents updates to these registers to protect against unauthorized changes to these pointers.

While SDMA is locked, the ONCE\_ENB register cannot be written to prevent the OnCE under ARM platform control from being used to gain access to SDMA internal memory. If ARM platform control of the OnCE is enabled before setting the LOCK bit, the ARM platform can use the ONCE for debug purpose after LOCK is set.

#### 41.4.5 OnCE and PCU Debug States

The SDMA has two different debug modes in which the OnCE performs debug instructions.

Refer to [Figure 41-4](#) for an example of the PCU states in debug. The following are the two debug states:

- When a channel is running (that is, when CCR and CCPRI are different from 0, which can be read in the PSW register), SDMA can execute a SoftBkpt instruction from the channel script or receive a debug request. When either happens, the SDMA enters its "Classical" *Debug* state, which is described in [OnCE and Real-Time Debug](#).
- When a channel is not running, the SDMA can be in *Sleep* state or in *Sleep after Reset* state. If a debug request is sent to the core, it enters its *Debug in Sleep* state. This debug mode works similarly to the "Classical" *Debug* state, except it returns to the original state (*Sleep* or *Sleep after Reset*) when the debug mode is left via the exec\_core instruction of the OnCE. From this *Debug in Sleep* state, the SDMA can execute a program whereas no channel is running. If a new debug request is sent to the core or if a SoftBkpt is executed, it comes back to this *Debug in Sleep* state.

The OnCE is provided with several instructions that can be executed when the core is in either debug state. The following table summarizes the behavior of these OnCE debug instructions. There exists other secondary OnCE instructions that are described in [OnCE and Real-Time Debug](#).

**Table 41-7. SDMA in Debug Mode**

Instruction	Debug	Debug in Sleep
exec_once	exec_once <instruction>  SDMA executes the <instruction> and returns to the <i>Debug</i> state. The Program Counter (PC) is not incremented. This command must not be used with an instruction that modifies the PC value.	exec_once <instruction>  SDMA executes the <instruction> and returns to the <i>Debug in Sleep</i> state. The Program Counter (PC) is not incremented. This command must not be used with an instruction that modifies the PC value.
run_core	run_core <instruction>  SDMA executes the <instruction>, leaves the <i>Debug</i> state and continues executing the channel script from the position where it stopped. This command must not be used with an instruction that modifies the PC value.	run_core <instruction>  SDMA executes the <instruction> and returns to its <i>Sleep</i> or <i>Sleep after Reset</i> initial state. This command must not be used with an instruction that modifies the PC value.
exec_core	exec_core <instruction>  It is similar to run_core except it requires an instruction that changes the PC value (jump, branch...): the SDMA jumps to the new PC value, leaves the <i>Debug</i> state and starts executing instructions from this new PC value.	exec_core <instruction>  If the previous state was <i>Sleep after Reset</i> , the SDMA returns to this state, and Chn0Addr value overrides the PC value.  Otherwise, the SDMA jumps to the new PC value and starts executing instructions from this new PC.

## NOTE

The feature exec\_core in *Debug in Sleep* after *Sleep after Reset* was added for the Channel boot (channel 0) to allow the debugger to return to *Sleep after Reset* state with a new PC

value. The SDMA will be ready to boot at the Chn0Addr address.

#### 41.4.6 SDMA Clocks and Low Power Modes

The SDMA receives several root clocks from the SoC clock controller block and performs adaptive clock gating to optimize its power consumption. From a user standpoint, clock gating and power mode selection are fully automatized inside the SDMA.

Root clock control is available from the SoC clock controller block.

There are numerous clock sources that are used in the SDMA. They belong to one of two possible clock domains listed in the following table, and have frequency constraints within each domain. Clocks are considered asynchronous between domains.

Within the ARM platform/SDMA clock domain, all clocks must come from the same DPLL. The ARM platform DMA interfaces (peripheral DMA and burst DMA) receive their clock from the ARM platform DMA clock source whose frequency can be once or twice the frequency of the SDMA core clock. The DMA interfaces are designed to work at the ARM platform DMA frequency, but the SDMA core is physically limited to a maximum 104 MHz frequency. Since this is lower than the maximum ARM platform DMA frequency, the SDMA core clock is tied to the ARM platform peripheral clock frequency.

The ARM platform Peripheral Bus Clock source must be an exact sub-frequency of the SDMA Core clock source (any integer value greater or equal to 1).

**Table 41-8. Clocking Scheme**

Clock Domain	Source Clock	Comments
ARM platform	SDMA core (SDMA main core)	Source clock for the core and all its operations; this clock is thus used by most of the SDMA sub-blocks.
	ARM platform DMA	DMA interface for the peripheral DMA and the burst DMA. It is balanced with the main clock source, and its frequency is either once or twice the main clock frequency.
	ARM platform peripheral	Connection to the ARM platform peripheral bus. It is a sub-frequency of the main clock frequency.
JTAG	TCK	Clock for JTAG access, limited to maximum of 1/8 of the SDMA core clock frequency.

The JTAG clock is sampled by the SDMA main clock to determine its rising edge. This simplifies design and clock management, but it also adds a ratio constraint between those two clocks. It is guaranteed the JTAG interface works properly when the frequency of TCK is lower than 1/8<sup>th</sup> of the frequency of the SDMA main clock (which is about 8 MHz when the SDMA core clock frequency is 66 MHz).

#### 41.4.6.1 Clock Gating and Low Power Modes

The SDMA automatically performs power saving without requiring user involvement. It implements two levels of automatic clock gating.

##### 41.4.6.1.1 Coarse Clock Gating

Every sub-block clock comes from one of the five available sources, and is gated with the sub-block specific enabling condition.

The following table displays the sub-block clocks and their source. It also indicates the relationships that may exist between different sub-blocks clock enables.

**Table 41-9. Sub-blocks Clocks**

Sub-block	Source Clocks	Enabling Condition and Comments	Related Enabling Conditions
Core	SDMA Main Core	The core sub-block clock is running when the core is not in one of its sleep states (Sleep or Sleep after Reset) or there is a pending channel. Typically, the core sub-block clock is stopped once all the channels are processed and the core enters its sleep state. A new pending channel awakes the core sub-block clock.	None
Memories	SDMA Main Core	The clock activation only occurs during a core access.	Disabled when Core sub-block clock is disabled or no memory access in progress
Scheduler	SDMA Main Core	Its clock only runs when scheduling is needed: for example, when there are pending channels, upon reception of a DMA request, and anytime the ARM platform modifies the channel running conditions.	None
ARM platform Control	SDMA Main Core & ARM platform peripheral	The ARM platform peripheral clock is solely used to determine the frequency ratio with the SDMA main clock. The control registers' clock is based on <i>SDMA main clock</i> ; it is active when the ARM platform or the SDMA modifies the contents of one of these registers.	None

*Table continues on the next page...*

**Table 41-9. Sub-blocks Clocks (continued)**

Sub-block	Source Clocks	Enabling Condition and Comments	Related Enabling Conditions
Burst DMA	SDMA Main Core & ARM platform DMA	The burst DMA has two clocks: The first clock is derived from the SDMA main core clock and drives registers that are connected to the FUBUS. The second clock is derived from the ARM platform DMA clock and drives registers that are connected to the ARM platform DMA bus outside the SDMA. Both clocks are enabled during active phases of data transfers (for example, these clocks are turned off when the burst DMA is not used by the running channel script).	Disabled when Core sub-block clock is disabled
Peripheral DMA	SDMA Main Core & ARM platform DMA	The peripheral DMA has two clocks: The first clock is derived from SDMA main clock and drives registers that are connected to the FUBUS. The second clock is derived from the ARM platform DMA clock and drives registers that are connected to the ARM platform DMA bus outside the SDMA. Both clocks are enabled during active phases of data transfers (for example, these clocks are turned off when the peripheral DMA is not used by the running channel script).	Disabled when Core sub-block clock is disabled
OnCE	SDMA Main Core	The OnCE clock is derived from main source clock. It is disabled by default. In order to use the OnCE, its clock must be explicitly turned on, either by enabling the OnCE access from the ARM platform peripheral bus (register ONCE_ENB), or by driving the clk_gating_off input pin high. This is a SDMA input whose driver depends on the SoC implementation (typically a JTAG controller).  The OnCE also receives the TCK input, which is the JTAG clock. It does not use it as a functional clock; the TCK input is sampled instead. Refer to <a href="#">Synchronization Implementation</a> .	When enabled, all other clocks are systematically on (clock gating is off)

#### 41.4.6.1.2 Refined Clock Gating

The SDMA implements a second level of clock gating on a register-per-register basis.

Unlike the first level that covers all the SDMA flip-flops, except the synchronizers (only five flip-flops are always running), the second level is only available for eligible registers, which amounts to about 90% of the SDMA flip-flops.

These gated registers are only clocked when the hardware logic detects a new data loading. This additional gating further reduces dynamic power consumption.

#### 41.4.6.1.3 Low Power Modes and User Control

Power savings are automatically managed by the SDMA hardware without any user involvement; however, one can distinguish three different power modes: SLEEP, RUN, and DEBUG.

The following table describes these modes, and shows how to switch from one mode to another.

**Table 41-10. Power Modes**

Power Mode	Sub-blocks								Comments
	Core	Mem ories	Sch edul er	ARM platf orm Cont rol	CRC	Burs t DMA	Peri pher al DMA	OnC E	
SLEEP	off <sup>1</sup>	off	wait <sup>2</sup>	wait	off	off	off	off	Set when the PCU state is either <i>Sleep</i> or <i>Sleep after Reset</i> and the SDMA is not in DEBUG mode. This is the default mode after reset.
RUN	on <sup>3</sup>	wait	wait	wait	wait	wait	wait	off	Set for the other PCU states that are reachable out of debug: <i>Program</i> , <i>Data</i> , <i>Change of Flow</i> , <i>Error in Loop</i> , <i>Debug</i> , <i>Functional Unit</i> , <i>Save</i> , or <i>Restore</i> .
DEBUG	on	on	on	on	on	on	on	on	Set regardless of the PCU state when clock gating is turned off to use the OnCE features (either <i>clk_gating_off</i> pin high or ONCE_ENB[0] set).

1. *off*: no clock
2. *wait*: only clocked when accessed or stimulated
3. *on*: clock is always running

It is possible to control the SDMA power mode. The procedures to force the SDMA into either mode are described in [SLEEP Mode](#).

#### 41.4.6.1.3.1 SLEEP Mode

This is the default mode after reset; therefore, resetting the SDMA forces this mode.

However, the common procedure is as follows:

- Ensure the *clk\_gating\_off* pin is low and ONCE\_ENB[0] is cleared.
- Disable all channels (via the STOP\_STAT control register, and the HO, DO, EO if necessary).
- Wait for the active channels to complete or force a reschedule via the reschedule bit in the RESET register.
- The SDMA is in SLEEP mode making it possible to completely shut off its clock from the chip level clock controller using the procedure described in [Stop Mode Response](#).

#### 41.4.6.1.3.2 RUN Mode

This is the default mode when a channel is running:

- Ensure the *clk\_gating\_off* pin is low and ONCE\_ENB[0] is cleared.
- Activate at least one channel (via the HSTART control registers, a DMA request, and/or the HO, DO, EO register bits).

#### **41.4.6.1.3.3 DEBUG Mode**

The DEBUG mode must be set when one needs to use the debugging facilities of the SDMA.

- Ensure the SDMA clocks are running from the CCM.
- Set the *clk\_gating\_off* pin high or use the SDMA to set ONCE\_ENB[0].

#### **41.4.6.1.4 Stop Mode Response**

The SDMA receives a stop request from the chip level clock controller. This request may be asserted when the chip enters the stop low power mode.

If the SDMA is running when the request is received, then the SDMA will complete all pending channels before returning to the SLEEP state. The SDMA sends an acknowledgement to the clock controller when the SLEEP state is entered indicating that the SDMA's clocks can be turned off.

### **41.4.6.2 Reset**

After reset (either received from the reset block or a software reset required by the ARM platform), the SDMA is in IDLE mode. It will start its boot code located at address 0 once a channel is activated.

Activating a channel can be done by the ARM platform after programming a positive priority and setting the channel bit in the EVTPEND register.

There will not be a context RESTORE for the first channel (bootload channel) called after a reset because the context data in RAM has not been initialized. Static context mode should be used for the first channel called after reset to ensure that the all context RAM for that channel is initialized. Subsequent calls to the same channel or different channels may use any of the dynamic context modes

## **41.4.7 Software Interface**

Appendix A fully describes the SDMA Application Programming Interface (API).



## 41.4.8 Initialization Information

This section discusses the following:

- [Hardware Reset](#)
- [Channel Script Execution](#)
- [Initialization and Script Execution Setup Sequence](#)

### 41.4.8.1 Hardware Reset

After reset, the program RAM, context RAM, data RAM, and RAM containing the channel enable registers (CHNENBLn) have unpredictable contents.

The active register set is assigned to channel 0 and the PC is initialized to all zeros. However, since the channel enable register is all zeros, there are no active channels and the SDMA is halted waiting for the boot channel to start.

The ARM platform will have to setup the SDMA in order to boot it. The CONFIG register must be initialized to determine the DMA/core clock ratio (1 or 2). Channel Enable Registers must also be initialized.

To start up the SDMA, the ARM platform first creates some channel control blocks (CCB) and buffer descriptors (BD) in ARM platform memory for the boot channel (channel 0) and then initializes the channel 0 pointer register (SDMA\_MC0PTR) to the address of the first control block. [Data Structures for Boot Code and Channel Scripts](#) provides an overview of the data structure for the CCB and BD's. The SDMA\_HSTART, SDMA\_HOSTOVR and SDMA\_EVT OVR registers are then configured according to [Runnable Channels Evaluation](#) to allow channel 0 to run.

Upon being enabled, the SDMA begins executing the script located at the address indicated by the Channel 0 Boot Address register (SDMA\_CHN0ADDR) in the program memory. The reset value of SDMA\_CHN0ADDR points to the default bootloader script in ROM. This ROM script will read the channel 0 pointer register (SDMA\_MC0PTR) to determine the location of the Channel Control Block (SDMA\_CCB) in ARM platform memory. The script will then begin fetching by DMA the first channel control block which contains a pointer to the location channel 0 Buffer Descriptor chain which is also fetched via DMA. If the buffer descriptor contains a valid command, the script interprets the command in each buffer descriptor and proceeds to implement the command and move on to the next buffer descriptor control block. The buffer descriptor commands for

channel zero are typically set up to load SDMA's program RAM, Data RAM, and initial values for the channel contexts. Some channel scripts expect particular parameters to be passed

There are two ways to make the SDMA boot on a user-defined script. The OnCE (either via its JTAG interface or its ARM platform Control interface) can be used to download any code in the SDMA RAM and force the SDMA to boot on that code. Also, the SDMA\_CHN0ADDR register in the ARM platform programming model can be modified to point to user code in RAM which would need to either have been loaded via the ONCE or default bootload routine (ex before a S/W reset).

### **41.4.8.2 Channel Script Execution**

The execution of an SDMA script depends on both the instructions that make up the script, the data context upon which it operates, and commands or parameters allowed to the buffer. All these items must be initialized before the script is allowed to execute.

Each of the 32 channels has a separate context, but may share scripts and locations in data RAM.

Channels are initialized by the ARM platform by using channel 0 to download any required scripts and data values and the channels initial context. The context contains all the initial values of the SDMA core registers. This includes the Program Counter (PC) which is set to the start of the desired script in SDMA program memory.

The ARM platform selects which trigger conditions that must occur for the channel to start by configuring the SDMA\_CHNENBL, SDMA\_HOSTOVR and SDMA\_EVT OVR registers. The trigger events include ARM platform setting HE (SDMA\_HSTART) or a hardware DMA request asserts an event input to SDMA. The channel can become active according to its priority compared with other runnable channels when the selected trigger(s) cause the condition described in [Runnable Channels Evaluation](#) to evaluate as true.

The specific parameters to be passed to each script in the buffer descriptor or context are documented in the software documentation for each script. Please refer to [SDMA Scripts](#) for complete script documentation. [Buffer Descriptor Format](#) provides an overview of the buffer descriptor format.

### **41.4.8.3 Initialization and Script Execution Setup Sequence**

To summarize, the following steps are minimally required to setup SDMA and run channel scripts.

- Perform Hardware Reset. The program RAM, context RAM, data RAM and SDMA\_CHNENBLn registers have unpredictable contents after this reset.
- Initialize SDMA\_CHNENBLn registers to map DMA request events to desired channels.
- Configure SDMA\_CHNPRIn registers to select priority for runnable channels. A non-zero priority is required for the channel to run.
- Configure the SDMA\_CONFIG register to select DMA to SDMA core clock ratio .
- Set up channel control blocks and buffer descriptors in ARM platform to specify the loading of SDMA program RAM and channel contexts for each SDMA channel to be used. Reference [Data Structures for Boot Code and Channel Scripts](#).
- Configure SDMA\_MC0PTR register with base address of ARM platform Channel Control Block base address.
- Initialize SDMA\_CHNENBLn registers to map DMA request events to associated channel. Reference [Mapping DMA Requests to Pending Channels](#).
- Configure SDMA\_CHNPRIn registers to set priority for each channel to be run.
- For each channel to be run, configure SDMA\_HOSTOVR (HO) and SDMA\_EVT OVR (EO) registers to select which events (hardware and/or software trigger events) must occur for the channel to be runnable. Reference [Runnable Channels Evaluation](#).
- Set bit 0 of the SDMA\_HSTART register to set HE[0] and allow Channel 0 to run (assumes EO[0] and DO[0] were both set in previous step). This will cause SDMA to load the program RAM and channel contexts configured previously.
- Wait for Channel 0 to finish running. This is indicated by HI[0]=1 in the SDMA\_SDMA\_INTR register, or by optional interrupt to the ARM platform.
- Set the LOCK bit in the SDMA\_SDMA\_LOCK register to prevent un-authorized uploads of data to SDMA RAM.
- Additional channel scripts can now be run by enabling the selected software or hardware trigger event according to [Runnable Channels Evaluation](#).

### 41.4.9 SDMA Programming Model

This section describes the programming model for the SDMA RISC engine, including its processor, memory, and internal control registers.

All addresses are related to the internal SDMA memory map, which is completely different from the ARM platform memory maps. The ARM platform processor has no access to any hardware resource described, except when those resources are described in ARM Platform Memory Map and Control Register Summary. .

### 41.4.9.1 State and Registers Per Channel

The SDMA can be seen as a set of 32 identical devices that are able to perform one data transfer channel each. Only one channel can work at a time, but every channel state is available at any time.

This chapter lists the components of every channel state.

### 41.4.9.2 General Purpose Registers

Each channel has eight general purpose registers of 32 bits for use by scripts. General register 0 has a dedicated function for the loop instruction, but otherwise can be used for any purpose.

### 41.4.9.3 Functional Unit State

Each channel context has some state that is part of the functional units.

The specific allocation of this state is part of the functional unit definition that is described in [Burst DMA Unit Programming](#), [Peripheral DMA Unit Programming](#).

This state must be saved/restored on context switches.

#### 41.4.9.3.1 Program Counter Register (PC)

The PC is 14 bits. Since instructions are 16 bits in width and all memory in the SDMA is 32 bits in width, the low order bit of the PC selects which half of the 32-bit word contains the current instruction.

A low order bit of zero selects the most significant half of the word.<sup>1</sup>

#### 41.4.9.3.2 Flags

Each channel has the following four flags:

- The T bit reflects the status of some arithmetic and test instructions. It is set when the result of an addition or a subtraction is zero and cleared otherwise. It is also the copy of the tested bits. Finally, it can also be set when the loop counter (GReg0) reaches zero. When the last instruction of the hardware loop is an operation that can modify the T flag, its effect on T is discarded and replaced by the GReg0 status.

---

1. For example, big-Endian.

- Two additional bits, SF and DF, are used to indicate error conditions resulting from loading data sources and storing to destinations, respectively. Access errors set these bits, and successful transactions clear them. They can also be cleared by specific instructions (CLRF and loop). The source fault (SF) is updated by the loads LD and LDF; the destination fault (DF) is updated by the stores ST and STF.
- Access errors are caused by several conditions including writing to the ROM, writing to a read-only memory mapped register, accessing an unmapped address, or any transfer error received by a peripheral when it is accessed.

The SF and DF flags have a major impact on the behavior of the hardware loop: If SF or DF is set when starting a hardware loop and it is not masked by the loop instruction, the loop body will not be executed. Inside the loop body, if a load or store sets the corresponding SF or DF flag, the loop exits immediately. Testing the status of the T flag at the end of the loop (as well as testing both SF and DF) tells if the loop exited abnormally as any anticipated exit prevents GReg0 from reaching the zero value and thus setting the T flag. This is also valid if the fault occurs at the last instruction of the last loop.

- The last flag is the loop mode flag, LM, which is composed of two bits. The most significant bit indicates when the processor is currently operating in loop mode. It is set by the loop instruction and is cleared after execution of the last instruction of the last loop. The least significant bit is set when the program counter points to the last instruction of a loop on the last path. It is used for a channel that is restored with this configuration to know that the next program counter is EPC. As with the dynamic context switch GReg0, which indicates when the program must get out of the loop, it can be restored only on the last instruction of the loop. This, however, is too late to fetch the next instruction after the loop.

#### 41.4.9.3.3 Return Program Counter (RPC)

The RPC is 14 bits. It is set by the jump to the subroutine instructions and used by the return from the subroutine instructions.

Instructions are available to transfer its contents to and from a general register.

#### 41.4.9.3.4 Loop Mode Start Program Counter (SPC)

The SPC is 14 bits. It is set by the loop instruction to the location immediately following it.

### 41.4.9.3.5 Loop Mode End Program Counter (EPC)

The EPC is 14 bits. It is set by the loop instruction to the location of the next instruction after the loop.

### 41.4.9.4 Context Switching-Programming

Each channel has a separate context consisting of the eight general purpose registers and additional registers representing the state of the functional units.

The active registers and functional units contain the context of the active channel. The context of inactive channels is stored in SDMA RAM, which is part of the SDMA address space.

In a function of the selected context switching mode ([Context Switching](#)), modified registers by the program can be saved in the channel RAM space while the program is going on. In every cycle, a write access to the RAM is possible.

On a done or yield(ge) instruction, SDMA goes into "real" context switching. In one of the dynamic modes, modified registers not previously saved, as well as the PC-Loop registers, are stored into the context area of the channel that will be closed. The new PC-Loop registers are loaded from the context area of the new channel. All other registers are restored while the program is executed, giving priority to registers used by the decoded instruction. Therefore, in the best case, only the PC and Loop registers should be saved and restored during this context-switching phase, which only requires five SDMA cycles.

In static mode, the context switch stores all registers in the old channel RAM space, and restores all registers from the new channel RAM space. It requires 26 SDMA cycles.

The address of the context memory for channel  $i$  is  $CONTEXT\_BASE + 24*i$  or  $CONTEXT\_BASE + 32*i$  where  $CONTEXT\_BASE$  equals 0x0800. The table below presents the layout of a channel context in memory:

**Table 41-11. Layout of a Channel Context in Memory for SDMA**

OFFSET	31	30	29-16	15	14	13-0
0	SF	-	RPC	T	-	PC
1	LM		EPC	DF	-	SPC
2	GR0					
3	GR1					
4	GR2					
5	GR3					
6	GR4					
7	GR5					

*Table continues on the next page...*

**Table 41-11. Layout of a Channel Context in Memory for SDMA (continued)**

8	GR6
9	GR7
10	MDA (burst DMA)
11	MSA (burst DMA)
12	MS (burst DMA)
13	MD (burst DMA)
14	PDA (peripheral DMA)
15	PSA (peripheral DMA)
16	PS (peripheral DMA)
17	PD (peripheral DMA)
18	
19	
20	Reserved <sup>1</sup>
21	Reserved <sup>1</sup>
22	Reserved <sup>1</sup>
23	Reserved <sup>1</sup>
24	Scratch RAM (optional)
25	Scratch RAM (optional)
26	Scratch RAM (optional)
27	Scratch RAM (optional)
28	Scratch RAM (optional)
29	Scratch RAM (optional)
30	Scratch RAM (optional)
31	Scratch RAM (optional)

#### 41.4.9.5 Address Space

The SDMA has four internal buses which are listed here.

- The Instruction bus reads instructions from the memory. Its address map is described in [Instruction Memory Map](#).
- The Data bus (DMBUS) accesses the same memories as those visible on the Instruction bus, some memory-mapped registers (scheduler status and OnCE registers), and up to 14 peripherals. Its address map is described in [Data Memory Map](#).



- The Functional Units bus (FUBUS) accesses the , Burst DMA, Peripheral DMA . The addressing mechanism is further detailed in [Functional Units Programming Model](#).
- The Context Switch bus reads/writes registers into context-switch RAM space. It is a 64-bit bus dedicated for accessing this RAM space for updating the context of the running channel. While the program is going on, this bus has the lowest priority compared to the Instruction and Data buses, except for restoring a register needed for the decoded instruction to be executed. On the save part of a context switch (when the PCU is in its slave state), this is the only one used. On the restore part, the Instruction bus has the priority to read the next instruction at the restored PC and otherwise the Context Switch bus is used. It is not possible to control the actual data transfers that occur on this bus.

#### 41.4.9.5.1 Instruction Memory Map

The instruction memory map is based on a 14-bit address bus and a 16-bit data (instruction) bus. Each address corresponds to a 16-bit data location.

Instructions are fetched from either program ROM or program RAM. An SDMA script is able to change the contents of the program RAM, which is also visible from the data bus.

The first two instruction locations (at 0 and 1) are special. Location 0 is where the PC is set on reset. Location 1 is where the PC is set upon the execution of an illegal instruction. It is expected that both of these locations will contain a jmp to handle routines.

**Table 41-12. SDMA Instruction Memory Space**

Device	SDMA Address (Hex)	Base Address Label	Block Name	WS	Description
ROM	0x0000 ↓ 0x07FF	SDMA_IBUS_ROM_ADDR	-	0	4 Kbyte internal ROM with boot code and standard routines.
RAM	0x1000 ↓ 0x1FFF	SDMA_IBUS_RAM_ADDR	-	0	8 Kbyte internal RAM with channels context and user data/routines.

#### 41.4.9.5.2 Data Memory Map

All of the data accessible to SDMA scripts make up the data memory space of the SDMA.

This address space has several components:

- ROM (also visible on the Instruction bus)
- RAM (also visible on the Instruction bus)



- Shared Peripherals Registers
- SDMA Internal Registers (scheduler, OnCE, and registers that are also accessible by the ARM platform)

SDMA scripts can read and write to the context RAM, data RAM, shared peripheral registers, and internal registers.

The address range is 16 bits and the data width is 32 bits. Each address corresponds to a 32-bit data word. When accessing peripheral registers (USB and so on), the data width may be different. The exact address map for the peripherals depends on the project (as presented in each respective chapter).

Data access is performed with *ld* and *st* instructions that take the address from a general purpose register in the core (GRegn). The mapping between the general purpose register contents and the address bus is given in the following table:

**Table 41-13. GRegn to DMBUS Address Mapping**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
sz	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
address															

Grayed bits are simply discarded but they must be cleared to ensure forward-script compatibility.

- sz (bit 31) indicates the peripheral data width: 0 is used for a 32-bit peripheral and 1 is used for a 16-bit peripheral.
- address (bits 15 down to 0) is the address of the accessed resource (internal memory, internal register, or shared peripheral).

**Table 41-14. SDMA Data Memory Space**

Device	SDMA Address (Hex)	Size	Description
ROM	0x0000 → 0x03FF	4 Kbyte	4 Kbyte internal ROM with boot code and standard routines
Reserved	0x0400 → 0x07FF	4 Kbyte	4 Kbyte Reserved
RAM	0x0800 → 0x0FFF	8 Kbyte	8 Kbyte internal RAM with channels contexts and user data/routines
per1	0x1000 → 0x1FFF	16 Kbyte	<i>peripheral 1</i> memory space (4 Kbyte peripheral's address space)
per2	0x2000 → 0x2FFF	16 Kbyte	<i>peripheral 2</i> memory space (4 Kbyte peripheral's address space)
per3	0x3000 → 0x3FFF	16 Kbyte	<i>peripheral 3</i> memory space (4 Kbyte peripheral's address space)
per4	0x4000 → 0x4FFF	16 Kbyte	<i>peripheral 4</i> memory space (4 Kbyte peripheral's address space)
per5	0x5000 → 0x5FFF	16 Kbyte	<i>peripheral 5</i> memory space (4 Kbyte peripheral's address space)
per6	0x6000 → 0x6FFF	16 Kbyte	<i>peripheral 6</i> memory space (4 Kbyte peripheral's address space)
Registers	0x7000 → 0x7FFF	16 Kbyte	Memory mapped registers

*Table continues on the next page...*

**Table 41-14. SDMA Data Memory Space (continued)**

Device	SDMA Address (Hex)	Size	Description
per7	0x8000 → 0x8FFF	16 Kbyte	<i>peripheral 7</i> memory space (4 Kbyte peripheral's address space)
per8	0x9000 → 0x9FFF	16 Kbyte	<i>peripheral 8</i> memory space (4 Kbyte peripheral's address space)
per9	0xA000 → 0xAFFF	16 Kbyte	<i>peripheral 9</i> memory space (4 Kbyte peripheral's address space)
per10	0xB000 → 0xBFFF	16 Kbyte	<i>peripheral 10</i> memory space (4 Kbyte peripheral's address space)
per11	0xC000 → 0xCFFF	16 Kbyte	<i>peripheral 11</i> memory space (4 Kbyte peripheral's address space)
per12	0xD000 → 0xDFFF	16 Kbyte	<i>peripheral 12</i> memory space (4 Kbyte peripheral's address space)
per13	0xE000 → 0xEFFF	16 Kbyte	<i>peripheral 13</i> memory space (4 Kbyte peripheral's address space)
per14	0xF000 → 0xFFFF	16 Kbyte	<i>peripheral 14</i> memory space (4 Kbyte peripheral's address space)

## 41.4.10 SDMA Initialization

Appendix A describes the setup of the SDMA . This section provides a quick description of several initialization procedures.

### NOTE

There may be differences with the actual implementation in the API.

### 41.4.10.1 Hardware Reset-SDMA

After reset, the RAM that holds contexts, data, scripts, and the DMA request-channels matrix has unpredictable content.

The core registers are all reset to 0, including the PC; the PCU state is *Sleep after Reset*. No channel can be activated because all of the priorities are also reset to 0.

### 41.4.10.2 Standard Boot Sequence

The following is the standard boot sequence:

1. Initialize the CONFIG register-detailed in [Configuration Register \(SDMAARM\\_CONFIG\)](#)-to determine the ARM platform DMA/core clock ratio (1 or 2)
2. Initialize the DMA request-channels matrix (see [Channel Enable RAM \(SDMAARM\\_CHNENBL<sub>n</sub>\)](#) ).
3. Program the channel control registers-[Channel Event Override \(SDMAARM\\_EVTOVR\)](#), [Channel BP Override \(SDMAARM\\_DSPOVR\)](#), [Channel](#)

BP Override (SDMA\_HOSTOVR), and [Channel Event Pending \(SDMAARM\\_EVTPEND\)](#)-according to the channel allocation.

4. Perform any necessary setup as required by the standard boot script in ROM (this is described in Appendix A).
5. Trigger channel 0 with the [Channel Start \(SDMAARM\\_HSTART\)](#) register, which starts the execution of the ROM script starting at address 0. This boot downloads channel scripts and contexts in RAM.

### 41.4.10.3 User-Defined Boot Sequence

The following is a user-defined boot sequence:

1. Initialize the [Configuration Register \(SDMAARM\\_CONFIG\)](#)[Channel Enable RAM \(SDMAARM\\_CHNENBL \$n\$ \)](#), [Channel Event Override \(SDMAARM\\_EVTOVR\)](#), [Channel BP Override \(SDMAARM\\_DSPOVR\)](#), [Channel ARM platform Override \(SDMAARM\\_HOSTOVR\)](#), and [Channel Event Pending \(SDMAARM\\_EVTPEND\)](#).
2. Use the OnCE (either via its JTAG interface or its ARM platform control registers) to download any code in the SDMA RAM. [Accessing the Memory](#) describes how to write data to the RAM via the OnCE.
3. Use the OnCE instructions to make the PC default value point to the new boot script start address, or rely on the ROM startup script, which first jumps to the address in [Channel 0 Boot Address \(SDMAARM\\_CHN0ADDR\)](#). (This register default address points to the standard boot script.)

### 41.4.10.4 Script Loading and Context Initialization

The execution of an SDMA script depends on both the instructions that make up the script and the data context upon which it operates. Both must be initialized before the script is allowed to execute.

Each of the 32 channels has a separate data context, but may share scripts and locations in the data RAM.

The ARM platform manages the space in program RAM and data RAM. It also manages the assignment of SDMA channels to the device drivers that need them. Channels are initialized by the ARM platform via the channel 0 boot script. The boot channel downloads any required scripts with their data and the channels' initial contexts. Every context contains all the initial values of the registers, including the PC. Then the ARM platform can enable any channel that becomes active and begins fetching and executing instructions from its script.

## 41.4.11 Instruction Description

The following sections introduce the instruction of the SDMA.

Instruction set details are available in [Instruction Set](#).

### 41.4.11.1 Scheduling Instructions

The following are scheduling instructions:

- **done**-The instruction causes certain scheduling or interrupt bits to be set or cleared, which may cause a change in the schedule-ability of the running channel. Then the instruction causes the SDMA to evaluate the current scheduling priorities and to choose the highest priority ready channel. If this channel is not the current channel, a context switch will take place. If there are no runnable channels, the SDMA will enter the stopped mode. The done 5 has a special usage reserved for debug, as explained in [Debug Instructions](#).
- **yield**-These instructions are special cases of the done instruction. They do not modify the scheduling bits, but allow the highest pending channel (if it exists) to preempt the current channel if the pending channel priority is strictly greater than the current channel priority.
- **yieldge**-These instructions are special cases of the done instruction. They do not modify the scheduling bits, but allow the highest pending channel (if it exists) to preempt the current channel if the pending channel priority is strictly greater or equal to the current channel priority.
- **notify**-The notify instruction affects the scheduling bits, but does not cause rescheduling.

### 41.4.11.2 Conditional Branch Instructions

The conditional branch instructions of an 8-bit displacement, which is sign-extended and added to the current PC (which points to the next instruction) if the condition is satisfied.

Otherwise, control passes to the next sequential instruction.

- **BF**-Branch if False. The branch is taken if the T bit in the processor status is zero (false).
- **BT**-Branch if True. The branch is taken if the T bit in the processor status is one (true).

- **BSF-Branch if Source Fault.** The branch is taken if the SF bit in the processor status is one.
- **BDF-Branch if Destination Fault.** The branch is taken if the DF bit in the processor status is one.

### 41.4.11.3 Unconditional Jump Instructions

There are two varieties of unconditional control transfers: an absolute transfer and a through-register transfer.

Absolute transfers have a 14-bit address field that replaces the current PC.

- **JMP-Jump.** Causes the processor to jump to an absolute address encoded in the instruction itself.
- **JSR-Jump to Subroutine.** Causes the processor to jump to a subroutine, the address of which is encoded in the instruction itself.
- **JMPR-Jump through Register.** Causes the processor to jump to an absolute address contained in a General register. This instruction is meant to be used when more than one level of subroutines are required.
- **JSRR-Jump to Subroutine through Register.** Causes the processor to jump to a subroutine, the address of which is contained in a General register. This instruction is meant to be used when more than one level of subroutines are required.

### 41.4.11.4 Subroutine Return Instructions

The following are subroutine return instructions:

- **RET-Return from Subroutine.** The RET restores the contents of RPC to PC.
- **LDRPC-Load from RPC to Register.** The LDRPC instruction is meant to be used when more than one level of subroutines are required. It stores the contents of RPC in any General register.

### 41.4.11.5 Loop Instruction

The following is a loop instruction:

**LOOP-Enters Loop Mode.** Before entering loop mode, the loop instruction can optionally clear the fault flags (SF and/or DF) based on a 2-bit field in the instruction. This feature is linked to the fact that setting SF or DF in loop mode will cause an immediate exit of the loop.

### 41.4.11.6 Miscellaneous Instructions

The following are miscellaneous instructions:

- **CLRF**-Clear Fault Flags. This instruction clears any combination of SF and DF.
- **MOV r,s**-This moves data from GReg[s] to GReg[r].
- **LDI r,immediate**-This loads GReg[r] with a zero-extended immediate value.

### 41.4.11.7 Logic Instructions

The following are logic instructions:

- **XORr,s**-This performs an exclusive or between GReg[r] and GReg[s], and stores the result in GReg[r].
- **XORIr,immediate**-This performs an exclusive or between GReg[r] and a zero-extended immediate value, and stores the result in GReg[r].
- **ORr,s**-This performs an or between GReg[r] and GReg[s], and stores the result in GReg[r].
- **ORIr,immediate**-This performs an or between GReg[r] and a zero-extended immediate value and, stores the result in GReg[r].
- **ANDNr,s**-This performs an and between GReg[r] and the negated GReg[s], and stores the result in GReg[r].
- **ANDNIr,immediate**-This performs an and between GReg[r] and the negated zero-extended immediate value, and stores the result in GReg[r].
- **ANDr,s**-This performs an and between GReg[r] and GReg[s], and stores the result in GReg[r].
- **ANDIr,immediate**-This performs an and between GReg[r] and a zero-extended immediate value, and stores the result in GReg[r].

### 41.4.11.8 Arithmetic Instructions

Arithmetic instructions modify the T bit in the processor status according to the result of the operation. The T bit is set if the result is zero, otherwise it is cleared.

- **ADD r,s**-This performs the addition of GReg[r] and GReg[s], and stores the result in GReg[r].
- **ADDI r,immediate**-This performs the addition of GReg[r] and a zero-extended immediate value, and stores the result in GReg[r].

- **SUB r,s**-This performs the subtraction of GReg[s] from GReg[r], and stores the result in GReg[r].
- **SUBIr,immediate**-This performs the subtraction of a zero-extended immediate value from GReg[r], and stores the result in GReg[r].

#### 41.4.11.9 Compare Instructions

Compare instructions modify the T bit in the processor status according to the result of the operation. The T bit is set if the comparison is true, otherwise it is cleared.

##### NOTE

Only one version of the immediate form is implemented. Non-equality comparisons to immediate values will require two instructions.

- **CMPEQ r,s**-This sets T when registers GReg[r] and GReg[s] are equal.
- **CMPEQIr,immediate**-This sets T when register GReg[r] and the zero-extended immediate value are equal.
- **CMPLTr,s**-This sets T when register GReg[r] is less than and not equal to GReg[s]. The comparison is signed.
- **CMPHS r,s**-This sets T when register GReg[r] is greater than or equal to GReg[s]. The comparison is signed.

#### 41.4.11.10 Test Instructions

Test instructions modify the T bit in the processor status according to the result of the operation. The T bit is set if any bit in the result is one, otherwise it is cleared.

- **TSTr,s**-This performs an and between GReg[r] and GReg[s], and sets T if the result is not zero.
- **TSTIr,immediate**-This performs an and between GReg[r] and a zero-extended immediate value, and sets T if the result is not zero.

#### 41.4.11.11 Byte Permutation Instructions

These instructions shuffle the bytes in a register. For the purpose of describing these instructions, have the bytes in a register be numbered from the most significant as  $b_3$ ,  $b_2$ ,  $b_1$ ,  $b_0$ .

- **RORBr**-The rotate right byte. The result is  $b_0$ ,  $b_3$ ,  $b_2$ ,  $b_1$ .

- REVB<sub>r</sub>-The reverse bytes in word. The result is  $b_0, b_1, b_2, b_3$ .
- REVBLO<sub>r</sub>-The reverse, two low-order bytes. The result is  $b_3, b_2, b_0, b_1$ .

#### **41.4.11.12 Bit Shift Instructions**

The following are bit shift instructions:

- ROR1<sub>r</sub>-The rotate right 1 bit. This instruction does a circular right shift of 1 bit.
- LSR1<sub>r</sub>-The logical shift right 1 bit. This instruction shifts all bits to the right by 1. The high order bit is replaced by a 0.
- ASR1<sub>r</sub>-The arithmetic shift right 1 bit. This instruction shifts all bits to the right by 1. The high order bit is replaced by itself.
- LSL1<sub>r</sub>-The logical shift left 1 bit. This instruction shifts all bits to the left by 1. The low order bit is replaced by zero.

#### **41.4.11.13 Bit Manipulation Instructions**

- BCLR<sub>r,n</sub>-The bit clear is immediate; clears bit number  $i$  in register  $r$ .
- BSET<sub>r,n</sub>-The bit set is immediate; sets bit number  $i$  in register  $r$ .
- BTST<sub>r,n</sub>-The bit test is immediate; tests bit number  $i$  in register  $r$  (T becomes equal to the selected register bit).

#### **41.4.11.14 SDMA Memory Access Instructions**

All memory accesses are 32 bits.

Any memory location that is implemented with less than 32 bits (for example, peripheral registers) causes unimplemented bits to be read as 0s.

All memory accesses will cause either the SF or DF flags in the processor status to be set if they cause a fault.

What constitutes a fault, especially when accessing peripheral registers, is a property of the memory location.

- LDr,(b,d)-The load instruction creates an address by adding the displacement field (d) to the contents of the base register (b). The SDMA location at the resulting address is read and placed in the destination register (r).
- ST<sub>r</sub>,(b,d)-The store instruction creates an address in the same manner as the load instruction. The register (r) is stored in the SDMA location at the resulting address.



### 41.4.11.15 Functional Unit Instructions

The functional unit instructions have an 8-bit field that is placed on the functional unit bus.

Some of these bits are used to select which functional unit should be involved in the transfer. The remaining bits are decoded by the selected functional unit so their specific use depends on the functional unit. See [Functional Units Programming Model](#).

There are two functional unit instructions, as follows:

- LDFr,fub-The 8-bit field is placed on the functional unit bus and a read is issued to the selected functional unit. As a result of this instruction, the SF may be set in the processor status.
- STFr,fub-The 8-bit field is placed on the functional unit bus and a write is issued to the selected functional unit. As a result of this instruction, the DF may be set in the processor status.

### 41.4.11.16 Illegal Instructions

All instruction encodings that are illegal cause the following actions:

- The current PC (which points to one beyond the offending instruction) is put in the EPC register.
- The loop mode bit is cleared.
- The PC is set to the value stored in the [Illegal Instruction Trap Address \(SDMAARM\\_ILLINSTADDR\)](#) register (the default value is 0x0001).

ILLEGAL-Although any instruction other than those indicated in the SDMA specification will trigger the illegal instruction mechanism, the ILLEGAL instruction code is preferred as it will always be kept as *illegal* in the possible future versions of the SDMA core.

### 41.4.11.17 Debug Instructions

The following are debug instructions:

- SOFTBKPT-The software breakpoint instruction causes the core to stop and enter debug mode. The core can then be accessed and started by the OnCE debug block only.

- done 5-This instruction is used for debugging, as it copies the contents of the PCU registers and flags to the context memory. Information on this instruction is described in [Saving the Context](#).
- CpShReg-This instruction copies the context memory into the PCU registers and flags. Modifying the corresponding memory location before executing this instruction enables you to have the channel continue from a new instruction address. This instruction is described in [Restoring the Context](#).

## 41.4.12 Functional Units Programming Model

The functional unit instructions cause an 8-bit code, found in the low eight bits of the instruction, to be asserted on the functional unit control bus.

Some of these bits are used to select one of several functional units. Functional units which can be selected include SDMA registers such as MSA and MSD which are not mapped in the SDMA memory map, and are accessible only through the functional unit bus. These Functional Unit Registers are listed in the following table. In order to establish a programming convention, assume the selection bits are some number of the most significant bits of the 8-bit code. Furthermore, some number of the least significant bits is decoded by a given functional unit to establish the type of operation to perform.

**Table 41-15. Functional Unit Registers**

Functional Unit	Register	Register Name	Section/Page
<a href="#">Burst DMA Unit Programming</a>	SDMSA	Memory Source Address Register	<a href="#">Memory Source Address Register (MSA)</a>
	MDA	Memory Destination Address Register	<a href="#">Memory Destination Address Register (MDA)</a>
	MD	Memory Data Buffer Register	<a href="#">Memory Data Buffer Register (MD)</a>  (Write) <a href="#">Burst DMA Write (stf)</a>  (Read) <a href="#">Burst DMA Read (ldf)</a>
	MS	Memory State Register	<a href="#">State Register (MS)</a>

*Table continues on the next page...*

**Table 41-15. Functional Unit Registers (continued)**

Functional Unit	Register	Register Name	Section/Page
Peripheral DMA Unit Programming	PSA	Peripheral Source Address Register	Peripheral Source Address Register (PSA)
	PDA	Peripheral Destination Address Register	Peripheral Destination Address Register (PDA)
	PD	Peripheral Data Buffer Register	Peripheral Data Register (PD)  (Write) Peripheral DMA Write (stf)-Write Mode  (Read) Peripheral DMA Read (ldf)-Read Mode
	PS	Peripheral State Register	Peripheral State Register (PS)

More information regarding the functional units can be found in [Peripheral DMA Unit](#), and [Burst DMA Unit](#).

#### 41.4.12.1 Burst DMA Unit Programming

The DMA instructions control the DMA state machine and may cause a DMA cycle on the associated memory bus.

There are four registers associated with the burst DMA unit: a Memory Source Address register (MSA), a Memory Destination Address register (MDA), a Memory Data buffer (MD), and a state register (MS). The burst DMA has two different uses:

- A data transfer between External Memory Interface and SDMA general register
- A data transfer in copy mode where blocks of data are transferred from the source address to the destination address

##### 41.4.12.1.1 Memory Source Address Register (MSA)

The source address register contains the pointer into EXTMC memory associated with the next read data transfer. It has byte granularity.

Reading the register with the ldf instruction has no side effects, and gives the address value in the EXTMC memory of the next data that is read by the SDMA during an ldf MD instruction.

Writing the source address register has two side effects: If the prefetch bit is set, a DMA read cycle (8-word read access) is issued with the new address. Any data still located in the buffer is lost. If there is valid write data in the buffer, it is necessary to force the DMA to completely flush it out before modifying MSA to guarantee all the data is effectively written to memory.

The MSA register has two modes of programming:

- Frozen-In frozen mode, the MSA register is not modified after DMA accesses.
- Incremented (default mode)-In incremental mode, MSA is incremented by the number of bytes transferred during read cycles.

#### **41.4.12.1.2 Memory Destination Address Register (MDA)**

The destination address register contains the pointer into EXTMC memory associated with the next write data transfer. It has byte granularity.

Reading the MDA register with the ldf instruction has no side effects. It gives the address value in the EXTMC memory where the next SDMA data (stf r,MD instruction) is stored when MD FIFO is flushed.

Writing the destination address register has one side effect. Any data still located in the buffer is lost. If there is valid write data in the buffer, it is necessary to force the DMA to completely flush it out before modifying MDA to guarantee all the data is effectively written to memory.

The MDA register has two modes of programming:

- Frozen-In frozen mode, the MDA register is not modified after DMA accesses.
- Incremented (default mode)-The MDA register is incremented by the number of bytes transferred during write cycles.

#### **41.4.12.1.3 Memory Data Buffer Register (MD)**

The data buffer register consists of a bank of 36 bytes that behave like FIFO.

This FIFO stores the eight words received when a read burst is triggered by the DMA (DMA is in read mode).

The MD register is in write mode after a writing in MDA or after an stf MD instruction.

In that case, a burst write access is automatically triggered when there are more than eight words in MD. For bandwidth optimization, any transfers between DMA and the EXTMC controller are based on burst accesses.

An `ldf r,MD|SIZE` instruction that reads the data buffer may cause a DMA cycle, as follows:

- If there are less bytes in the FIFO than the size parameter of the instruction. For instance, if only two bytes are available in MD and a 4-byte read is requested, a burst read access is executed to complete the two bytes.
- If the prefetch bit is set, and after reading there is enough space in the FIFO to store a full burst, a burst read access is triggered.

An `stf r,MD|SIZE` instruction that writes to the data buffer may cause a DMA cycle if the number of written bytes in MD is higher than 32 (eight words) or if the flush bit is set.

When DMA is used for data transfer between SDMA and EXTMC (reading or writing), no immediate error is possible because the block manages a data misalignment issue; therefore, it is allowed to read/write a word to/from a half-word address. However, the addresses (source or destination) must belong to the EXTMC memory mapping. The only potential error, in this mode, would be the error sent back by the EXTMC controller when an access to a super-user page is detected. The whole transfer on the DMA associated bus will be considered successful when there are no errors seen on the bus during the transfer. In copy mode, an immediate error could be returned to SDMA as described in [Burst DMA Unit Error Management](#).

#### 41.4.12.1.4 State Register (MS)

The state register contains the DMA state-machine value. It can be accessed in case of an error received during a transfer. MS is also accessed to set-up the conditional yielding feature.

The initialization value of this register is 0 and it consists of the following:

**Table 41-16. SDMA\_MS Structure**

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	0	0	spriv	stype	0	0	dpriv	dtype
W																
R	0	0	0	0	y	d	e		0	0	n					
W																

**Table 41-17. SDMA\_MS Field Descriptions**

Field	Description
31-22	Reserved

*Table continues on the next page...*

**Table 41-17. SDMA\_MS Field Descriptions  
(continued)**

Field	Description
21 spriv	The spriv value is ignored for this device. 0 = valid value 1 = Reserved
20 stype	Source Mode. Indicates if MSA has to be incremented (or not) during accesses. 0 Frozen-MSA is not modified. 1 Incremented-MSA is incremented by the number of transferred bytes during read access.
19-18	Reserved
17 dpriv	The dpriv value is ignored for this device. 0 = valid value 1 = Reserved
16 dtype	Destination Mode. Indicates if MDA has to be incremented (or not) during accesses. 0 Frozen-MDA is not modified. 1 Incremented-MDA is incremented by the number of transferred bytes during write access.
15-12	Reserved
11 y	Conditional Yielding selector. When selected, theyield/yieldge instructions will not switch channels if the Burst DMA is in Write Mode, and it has less than four bytes in its FIFO. This is aimed at reducing the number of inefficient FIFO flushes due to context switches. 0 Always yields 1 Yields conditionally (when there are less than four bytes in the FIFO in write mode)
10 d	Access Direction or DMA Mode. DMA is in write mode when data was written into MD by stf MD instructions, or if a previous DMA cycle on the external bus was a write access. Writing MDA or MSA changes the DMA mode to the respective value. DMA is in read mode when a previous DMA cycle was a read access, and DMA stays in read mode when data is read by SDMA with an ldf MD instruction. Reading MDA or MSA does not change the DMA mode. 0 Read Mode 1 Write Mode
9-8 e	Error. Indicates if the previous access was acknowledged with a bus error. 00 No error was received. 01 <i>reserved</i> 10 Error mode 11 error read burst
7-6	Reserved
5-0 n	Number of bytes in the MD FIFO.

#### 41.4.12.1.5 Burst DMA Write (stf)

When received from a stf instruction, the function code bits are interpreted as follows, depending on the addressed register:

**Table 41-18. STF Code Bits**

Register	7	6	5	4	3	2	1	0
MSA	s		p	freeze	r			spriv
MDA								dpriv
MD			f	cpy			sz	
MS								

**Table 41-19. STF Code Bit Field Descriptions**

Field	Description
7-6 s	Functional Unit selector 00 for Burst DMA
5 p (MSA)	Prefetch Flag 0 No prefetch 1 Prefetch required from new MSA
5 f (MD)	Forced Flush Flag 0 Automatic flush 1 FIFO contents are flushed (including the new written data).
4 freeze (MSA/MDA)	Address Freeze Mode 0 Address is normally incremented. 1 Address is frozen.
4 cpy (MD)	Copy Mode selection 0 Write Mode 1 Copy Mode
3-2 r	Register selection 00 MSA 01 MDA 10 MD 11 MS
1-0 sz (MD/MS)	Transfer Size 00 size 0 (no data stored in the FIFO) 01 byte (8 bits) 10 half-word (16 bits) 11 word (32 bits)
0 spriv (MSA)	The spriv value is ignored for this device. 0 = valid value 1 = Reserved
0 dpriv (MDA)	The dpriv value is ignored for this device. 0 = valid value 1 = Reserved

The possible write instructions are listed in the table below (unused bits should always be cleared).

**Table 41-20. Burst DMA STF Instruction List**

Binary	Assembly	Comments
00_0_0_00_00	stf r,MSA	Writes content of the SDMA general register (r) to the source address register. MSA is in incremented mode.
00_0_1_00_00	stf r,MSAIFR	Writes content of the SDMA general register (r) to the source address register. MSA is in frozen mode.
00_1_0_00_00	stf r,MSAIPF	Writes content of the SDMA general register (r) to the source address register, and starts a read burst access. MSA is in incremented mode.
00_1_1_00_00	stf r,MSAIPFIFR	Writes content of the SDMA general register (r) to the source address register, and starts a read burst access.
00_0_0_01_00	stf r,MDA	Writes content of the SDMA general register (r) to the destination address register. MDA is in incremented mode.
00_0_1_01_00	stf r,MDAIFR	Writes content of the SDMA general register (r) to the destination address register. MDA is in frozen mode.
00_1_0_10_00	stf r,MDISZ0IFL	No data transfers between the SDMA and MD, but all valid written data of the MD is flushed to the memory. An acknowledge or error is sent back to the SDMA core on transfer completion.
00_0_0_10_01	stf r,MDISZ8	8-bit (byte) transfer to write buffer MD
00_1_0_10_01	stf r,MDISZ8IFL	8-bit (byte) transfer to write buffer MD and flush after transfer. All valid written data of the MD is flushed to memory.
00_0_0_10_10	stf r,MDISZ16	16-bit (half-word) transfer to write buffer MD
00_1_0_10_10	stf r,MDISZ16IFL	16-bit (half-word) transfer to write buffer MD and flush after transfer. All valid written data of the MD is flushed to memory.
00_0_0_10_11	stf r,MDISZ32	32-bit (word) transfer to write buffer MD
00_1_0_10_11	stf r,MDISZ32IFL	32-bit (word) transfer to write buffer MD and flush after transfer. All valid written data of MD is flushed to memory.
00_0_1_10_00	stf r,MDICPY	No data transfer between SDMA and MD but starts a copy transfer whose length is given by the 4 LSB of r register. (Maximum burst length is eight words.)
00_0_0_11_11	stf r,MS	32-bit (word) transfer to status register MS
00_0_0_11_00	stf r,MSISZ0	Clears the error flag (if set). Other MS bits are unchanged; this instruction is also known as cref MS.

### NOTE

When a flush bit is set, the SDMA flushes the FIFO including the newly written data. An acknowledge is sent to the core before the flush completes (except if size 0 is used). The goal of this flush bit is to force a flush, but it is recommended to use it only when needed (for example, when finishing a row of pixels during 2D data transfers). Indeed, if this bit is omitted and if there are more than 32 bytes in the FIFO, a burst write access is automatically triggered.



Since all the stf r,MD instructions (including the copy mode) acknowledge the SDMA core before the store is effective (except if size 0 is used), it is recommended to perform an ldf from MS before terminating a channel in order to check the final error status. (The ldf from MS will stall the core until all the data was flushed out and the transfer status is known.)

After every stf MD instruction, the MDA is incremented by the number of bytes that are written in MD, except when it is programmed in frozen mode.

#### 41.4.12.1.6 Burst DMA Read (ldf)

When received from an ldf instruction, the function code bits are interpreted as follows, depending on the addressed register:

**Table 41-21. LDF Code Bits**

Register	7	6	5	4	3	2	1	0
MSA	s				r			
MDA								
MD			p				sz	
MS								

**Table 41-22. LDF Code Bit Field Descriptions**

Field	Description
7-6 s	Functional Unit selector 00 for Burst DMA
5 p (MD)	Prefetch Flag 0 no prefetch 1 automatic prefetch
3-2 r	Register selection 00 MSA 01 MDA 10 MD 11 MS
1-0 sz (MD)	Transfer Size 00 reserved 01 byte (8 bits) 10 half-word (16 bits) 11 word (32 bits)

The table below lists the possible write instructions (unused bits should always be cleared).

**Table 41-23. Burst DMA LDF Instruction List**

Binary	Assembly	Comments
00_0_0_00_00	ldf r,MSA	Copies the source address register value into an SDMA general register. It gives the memory address of the next data that will be read with an ldf MD instruction.
00_0_0_01_00	ldf r,MDA	Copies the destination address register value into an SDMA general register. It gives the memory address where the next incoming data will be flushed.
00_0_0_10_01	ldf r,MDISZ8	8-bit (byte) read
00_1_0_10_01	ldf r,MDISZ8 PF	8-bit (byte) read. If after this reading and the MD FIFO is empty, a burst read access at the MSA address is triggered.
00_0_0_10_10	ldf r,MDISZ16	16-bit (half-word) read
00_1_0_10_10	ldf r,MDISZ16 PF	16-bit (half-word) read. If after this reading, and the MD FIFO is empty, a burst read access at the MSA address is triggered.
00_0_0_10_11	ldf r,MDISZ32	32-bit (word) read
00_1_0_10_11	ldf r,MDISZ32 PF	32-bit (word) read. If after this reading and the MD FIFO is empty, a burst read access at the MSA address is triggered.
00_0_0_11_00	ldf r,MS	Copy the status register value into an SDMA general register.

### NOTE

Read data is 0-extended before writing in the SDMA general registers. When reading the MD register, the DMA takes data from the FIFO if it is available. If part or whole data is not in the FIFO, an external burst read access is performed to provide the missing data. The SDMA is stalled as long as the required read data is not complete.

After every reading, MSA is incremented by the number of read bytes from MD FIFO, except when MSA is programmed in frozen mode.

#### 41.4.12.1.7 Prefetch/Flush and Auto-Flush Management-Burst DMA Unit

The prefetch and auto-flush management enables the SDMA RISC machine to go on while a DMA access is performed.

When the RISC core requires a prefetch ( $p = 1$ ) to the Burst DMA, it will receive an immediate transfer acknowledge before the DMA has finished the external access. This enables the RISC core to do other things like accessing another DMA machine.

The basic principle in prefetch mode is for the DMA to anticipate data reads from the SDMA RISC engine by fetching external bursts of data as soon as there is enough space in the DMA FIFO to store it. If ever the RISC engine required data that is not available in the FIFO, the read acknowledge is delayed until the data is available, but it does not have to wait until the burst completes.

The auto-flush basic principle is similar: An automatic flush is triggered every time there are eight words to be written in the FIFO. If the FIFO is full and the RISC engine requires another write, it is stalled until the burst has started and enough space was freed in the FIFO to store that new data. This means the SDMA RISC engine does not have to wait for the completion of a burst to receive its acknowledge and continue its processing.

In particular, an auto-flush is executed when DMA is in write mode and if the following is true:

- If the FIFO is empty and the first write is to a word-aligned address of any size (ex: the 2 LSB of MDA[1:0]= 0x0), the auto-flush is triggered immediately after the write of the 32'nd byte.
- If the FIFO is empty, and if MDA is an odd byte address (1, 3, 5, 7,...) and an stf MDISZ8 is executed, the byte is flushed to memory. Once MDA increments to a word aligned address, the auto-flush will be triggered every 32 bytes.
- If the FIFO is empty, and if MDA is a half-word address (2, 6, 0xA,...) and an stf MDISZ16 is executed, the two bytes of the incoming data are flushed to memory. Once MDA increments to a word aligned address, the auto-flush will be triggered every 32 bytes.
- If the FIFO is empty, and if MDA is not a word-aligned address (ex 1, 2, 3, 5, 6, 7, 9,...), and an stf MDISZ32 is executed, the first 1 to 3 bytes will be flushed up to the next word aligned address. Afterwards, an auto-flush will be triggered each time the FIFO receives 32-bytes.
- Therefore, if an stf MDISZ32 is executed with MDA equal to 0x1 and with an empty MD FIFO, the bytes located at addresses 1, 2, and 3 are flushed, and the byte located at address 4 remains in MD FIFO. This solves the misalignment issue. Additionally, the next write instructions (stf) complete the FIFO until it contains eight words; then a burst write is executed by the DMA to empty the FIFO. Protocol on the external bus does not support bursts of different data types (byte, half-word, or word).

For example, consider the case where data is written using a byte access, stf MDISZ8. The value of MDA during the very first byte write determines when the auto-flush will occur as follows:

- If MDA=0x0, the flush occurs following the write of byte 32
- If MDA=0x1, the flush occurs following the write of byte 1, byte 3 and byte 35.
- If MDA=0x2, the flush occurs following the write of byte 2 and byte 34.

- If MDA=0x3, the flush occurs following the write of byte 1 and byte 33.
- If MDA=0x4, the flush occurs following the write of byte 32

The flush command forces the DMA to flush all MD valid bytes to the EXTMC controller. An acknowledge is sent immediately to the SDMA, and any potential error is reported on a future access. It is thus essential to conclude a transfer with a last read from MS, which will stall the core until all data was flushed out and returned to the transfer status (acknowledge or error).

### NOTE

During this kind of auto-flush (which occurs only at the beginning of a misaligned write transfer) no acknowledge is sent back to the SDMA, which is stalled until a flush is completed.

## 41.4.12.1.8 Data Alignment and Endianness-Burst DMA Unit

### 41.4.12.1.8.1 Burst DMA in Read Mode

For every read access to MD, the data returned to the SDMA core and the new FIFO state depends on the MSA status and the access size.

The FIFO is considered as a stack of 36 bytes: Data is fetched externally on a 32-bit bus, but the valid bytes only are stored in the FIFO and left-aligned (for a transfer of consecutive words, it is only the first word that may be truncated). The following table shows the FIFO byte alignment strategy and the corresponding MSA, the returned data, and the new FIFO state for any access size of an internal read from MD.

**Table 41-24. FIFO Read Configuration**

Before read		Internal read access size	Read data	After read	
MSA[1:0]	FIFO state			MSA[1:0]	FIFO state
00	x0 x1 x2 x3 y0 y1 y2 y3 z0 z1 z2 z3 and so on...	sz8	00 00 00 x0	01	x1 x2 x3 y0 y1 y2 y3 z0
		sz16	00 00 x0 x1	10	x2 x3 y0 y1 y2 y3 z0 z1
		sz32	x0 x1 x2 x3	00	y0 y1 y2 y3 z0 z1 z2 z3

*Table continues on the next page...*

**Table 41-24. FIFO Read Configuration (continued)**

Before read		Internal read access size	Read data	After read	
MSA[1:0]	FIFO state			MSA[1:0]	FIFO state
01	x1 x2 x3 y0 y1 y2 y3 z0 z1 z2 z3 t0 and so on...	sz8	00 00 00 x1	10	x2 x3 y0 y1 y2 y3 z0 z1
		sz16	00 00 x1 x2	11	x3 y0 y1 y2 y3 z0 z1 z2
		sz32	x1 x2 x3 y0	01	y1 y2 y3 z0 z1 z2 z3 t0
10	x2 x3 y0 y1 y2 y3 z0 z1 z2 z3 t0 t1 and so on...	sz8	00 00 00 x2	11	x3 y0 y1 y2 y3 z0 z1 z2
		sz16	00 00 x2 x3	00	y0 y1 y2 y3 z0 z1 z2 z3
		sz32	x2 x3 y0 y1	10	y2 y3 z0 z1 z2 z3 t0 t1
11	x3 y0 y1 y2 y3 z0 z1 z2 z3 t0 t1 t2 and so on...	sz8	00 00 00 x3	00	y0 y1 y2 y3 z0 z1 z2 z3
		sz16	00 00 x3 y0	01	y1 y2 y3 z0 z1 z2 z3 t0
		sz32	x3 y0 y1 y2	11	y3 z0 z1 z2 z3 t0 t1 t2

**41.4.12.1.8.2 Burst DMA in Write Mode**

For every write access to the MD, the new FIFO state depends on the MDA status and the access size.

The FIFO is considered as a stack of 36 bytes: Data is stored in the FIFO according to the internal access size and the former MDA value. The following table shows the FIFO byte alignment strategy corresponding to MDA, as well as the new FIFO state for any access size of an internal write to MD.

**Table 41-25. FIFO Write Configuration**

Before write		Internal write access size	Written data	After write	
MDA[1:0]	FIFO state			MDA[1:0]	FIFO state
00	tt uu vv ww ?? ?? ?? ?? ?? ?? ?? ?? and so on...	sz8	?? ?? ?? x0	01	tt uu vv ww x0 ?? ?? ?? ?? ?? ?? ??
		sz16	?? ?? x0 x1	10	tt uu vv ww x0 x1 ?? ?? ?? ?? ?? ??
		sz32	x0 x1 x2 x3	00	tt uu vv ww x0 x1 x2 x3 ?? ?? ?? ??
01	tt uu vv ww xx ?? ?? ?? ?? ?? ?? ?? and so on...	sz8	?? ?? ?? x0	10	tt uu vv ww xx x0 ?? ?? ?? ?? ?? ??
		sz16	?? ?? x0 x1	11	tt uu vv ww xx x0 x1 ?? ?? ?? ?? ??
		sz32	x0 x1 x2 x3	01	tt uu vv ww xx x0 x1 x2 x3 ?? ?? ??
10	tt uu vv ww xx yy ?? ?? ?? ?? ?? ?? and so on...	sz8	?? ?? ?? x0	11	tt uu vv ww xx yy x0 ?? ?? ?? ?? ??
		sz16	?? ?? x0 x1	00	tt uu vv ww xx yy x0 x1 ?? ?? ?? ??
		sz32	x0 x1 x2 x3	10	tt uu vv ww xx yy x0 x1 x2 x3 ?? ??

Table continues on the next page...

**Table 41-25. FIFO Write Configuration (continued)**

Before write		Internal write access size	Written data	After write	
MDA[1:0]	FIFO state			MDA[1:0]	FIFO state
11	tt uu vv ww xx yy zz ?? ?? ?? ?? ?? and so on...	sz8	?? ?? ?? x0	00	tt uu vv ww xx yy zz x0 ?? ?? ?? ??
		sz16	?? ?? x0 x1	01	tt uu vv ww xx yy zz x0 x1 ?? ?? ??
		sz32	x0 x1 x2 x3	11	tt uu vv ww xx yy zz x0 x1 x2 x3 ??

**NOTE**

If the FIFO mode changes from a write to a read mode, all remaining written bytes in MD are lost but no error is returned. Typically, this happens if an ldf MD is executed after stf MD instructions. Before a mode change, it is recommended to force the flush of a potential remaining byte by a stfMD|SZ0|FL instruction. In the same way, if a FIFO mode changes from a read to a write mode, all prefetched data present in the FIFO is lost and no error is returned.

**41.4.12.1.8.3 Endianness-Burst DMA Unit**

Big and Little Endian are supported by the Burst DMA, but data is always stored in MD in Big Endian.

Byte manipulation is performed when data is exchanged with an Burst controller (for example, during read or write burst accesses).

**41.4.12.1.9 Burst DMA Unit Copy Mode**

A mechanism is available to perform fast ARM-to-ARM transfers.

Data does not flow through the SDMA core: It is kept in the DMA FIFO. This mechanism is selected when writing MD with a special option in the instruction code (copy flag).

It is possible to transfer up to eight words in one SDMA instruction (this does not mean in one cycle). In this mode, every time an stf MD|CPY is executed, a read burst is executed and directly followed by a write burst transfer. Burst transfers are limited to

## Functional Description

eight words. The size of the transfer (in words)-given by the SDMA general register (4 LSB)-is also limited to eight. The following SDMA code shows how 100 bytes could be copied from the MSA address to the MDA address. This is sample code only.

### Burst DMA copy mode example

```
ldi r0,@src
stf r0,MSA // Source address setup
ldi r1,@dst
stf r1,MSA // Destination address setup
ldi r0,0x64 // data transfer counter
ldi r1,0x8

MAIN_XFER:
cmphs r0,r1 // Is r0 >= 0x8
bf LAST_XFER // If not, jump to last transfer label
stf r1,MD|CPY // Copy 8 words from MSA to MDA address.
subi r0,0x8 // Decrement counter
jmp MAIN_XFER // return to main transfer loop

LAST_XFER:
stf r0,MD|CPY
```

The main transfer loop is executed 12 times; then r0 equals 4 and the last transfer loop is run.

In this mode, an acknowledge is transmitted to the core as soon as the read burst can start; thus, a first copy instruction returns an immediate acknowledge and subsequent copy instructions will be acknowledged as soon as the previous copy has finished.

#### 41.4.12.1.10 Burst DMA Unit Error Management

Another point to consider is the management of errors.

Because the DMA immediately sends an acknowledge to the RISC core (except for the stf MS|SZ0|FLS instruction), it assumes no error will occur. If an error occurs, it is flagged (transfer error acknowledge) for the following DMA access.

This should not be a problem if the DMA is used properly. The MD accesses are meant to stall the SDMA as little as possible to optimize throughput and hide calculation time. Therefore, final access to MS should be performed before closing a channel. This access waits until any pending operation is finished in the burst DMA and gather any remaining error.



In copy mode, an error could be immediately returned to the SDMA on execution of the ldf copy or stf copy instruction. It happens when MSA or MDA are not word addresses (for example, 0[4]). This is because copy mode must only be used for transferring a large packet of aligned data.

When an error is received during a *read* transfer to the external bus, which may occur during the burst accesses, the MD FIFO contains the valid beats of the burst, and the error flag of MS is set to 2'b11 (error read burst). It is possible to read MS ("n" field) to know how much valid data remains in MD and when MD is empty (after ldf instructions). The next read MD instruction sets the MS error flag to 2'b10 (error mode), and an error is sent back to the SDMA core. In error mode, it is possible to read MSA, which gives the address of the error data. Any attempt to read or write MD, or to modify MDA or MSA in error mode, gives rise to an error; therefore, an error flag must be reset by clearing MS at the end of the SDMA code section responsible for error management.

In "error read burst" mode, writing MDA, MSA, or MD, or starting a copy transfer by a stf MDICOPY instruction will cancel the error mode. The following table shows when an immediate error is sent back according to the executed instruction.

**Table 41-26. Possibilities in ERROR READ BURST Mode**

DMA Instruction	Immediate Error	Comments
stf rn, MD stf rn, MSA (IU IPF) stf rn, MDA stf rn, MDICOPY	NO	Error mode is reset. MSA, MDA, or MD are updated and a DMA cycle may start. For the stf MDICOPY, a copy loop is executed.
stf rn, MS	NO	MS is updated.
ldf rn, MS ldf rn, MSA ldf rn, MDA	NO	MS, MSA, and MDA could be read in ERROR READ mode without any side effects (for example, no DMA cycle is triggered).
ldf rn, MD	YES/NO	Immediate error if there is no more data available for read in the FIFO.

When an error is received during a *write* transfer, the error is reported to the next DMA access. In this case, an error is sent to the SDMA core and the DMA goes to its error mode. Reading MS gives the number of bytes that remain in MD; reading MDA gives the address of the error data. Any attempt to read or write MD, or to modify MDA or MSA in error mode, give rise to an error; therefore, an error flag must be reset by clearing MS at the end of the SDMA code section responsible for error management.

**Table 41-27. Possibilities in ERROR Mode**

DMA Instruction	Immediate Error	Comments
stf rn, MD stf rn, MSA stf rn, MDA	Yes	Any attempt to modify MD, MSA, MDA will raise an immediate error and burst DMA remains in error mode. When address registers are write-accessed, an error is returned.

*Table continues on the next page...*

**Table 41-27. Possibilities in ERROR Mode (continued)**

DMA Instruction	Immediate Error	Comments
stf rn, MS	No	This is the only way to exit error mode. MS[9:8] must be reset by an stf MSISZ0 instruction.
ldf rn, MS ldf rn, MSA ldf rn, MDA	No	MS, MSA, and MDA could be read in error mode without any side effects (for example, no DMA cycle is triggered).
ldf rn, MD	Yes	Whatever the DMA direction (read or write), an ldf rn triggers an immediate error.

#### 41.4.12.1.11 Conditional Yielding-Burst DMA Unit

The standard SDMA transfer is based upon a hardware loop that has the following structure:

##### Hardware Loop

```

loop
load Rn,source           // can be ldf or ld
<computation>           // can be done through functional units
store Rn,dest            // can be st or stf
done 0                   // yield

```

This structure needs to be kept independent of the functional units' particularities regarding the context switch. However, there can be variations in the context switch's efficiency, which can depend on the number of data received up to that point, and on the data itself.

The DMA, with its 8-word burst capability, has a preferable context switch period when its address register is 8-word aligned: It is the only moment that occurs once every eight loops when the succession of bursts is not broken by the context switch. When this is not the case, a context switch requires the storing (or loading) of less than eight words, which requires separate accesses and is far less efficient. The rest of the 8-word packet is stored (or loaded) after the context restore, and this is done as separate accesses.

The proposed solution is a conditional yielding, which occurs only when the DMA is in an optimum state. It does not require any modification to the scripts. The condition is decided at the DMA level.

The DMA can be programmed in two modes-conditional or always-true-for every channel, which provides complete flexibility. By default, the DMA is not in conditional mode.

The DMA condition is computed from the FIFO fill level and the various modes, as follows:

- When copy mode is selected, regardless of the transfer direction ('read' or 'write'), the condition is always true.
- In read mode, the condition is always true.
- In write mode, the condition is true when there are four bytes or less in the FIFO; it is false when there are more than four bytes. The 4-byte limit comes from the possibility of saving those bytes as MD with absolutely no impact on the bus accesses.

The aim at conditional yielding is to avoid splitting bus accesses (especially bursts).

### 41.4.12.2 Peripheral DMA Unit Programming

The peripheral DMA unit is connected to the Multi-Layer DMA Crossbar Switch of the ARM platform.

Its goal is to perform data transfers between any blocks connected to the DMA bus of this platform. These blocks are either peripherals or memories. The peripheral DMA could be seen as the ARM platform DMA controller.

The DMA performs data transfers in three modes:

- Read mode, where data is read from peripherals or from memory connected to the ARM platform and copied in a SDMA general register.
- Write mode, where data of a general register has to be written in a peripheral or a memory.
- Copy mode, where data is read from a peripheral (or memory) at a source address (PSA) and automatically written to a peripheral (or memory) at a destination address (PDA).

In copy mode, no SDMA general register is involved as transferred data only goes through the data register of the DMA.

The peripheral DMA has three addressing modes: frozen, incremented, and decremented, as follows:

- Frozen mode-When source or destination addresses are frozen, their value is not modified after a transfer. This mode is typically used for addressing peripheral FIFOs located at a fixed address.
- Incremented mode-When source or destination addresses are in incremented mode, after every transfer they are incremented by the number of bytes transferred.
- Decrement mode-In decremented mode, addresses are decremented by the number of bytes transferred.

The peripheral DMA registers are as follows:

- Two, 32-bit address registers (PSA and PDA) that respectively contain the source address for a read access and the destination address for a write access
- A 32-bit status register (PS) that contains information on the peripheral DMA configuration, such as the number of valid bytes in the data register, the error flag, the source and destination address mode, and so on.
- A 32-bit data register (PD) that stores data involved in a data transfer

#### **41.4.12.2.1 Peripheral Source Address Register (PSA)**

The source address register contains a pointer to a source peripheral or a memory associated with the next read data transfer. It has byte granularity.

It is based on the following:

- A 32-bit register (PSA) to store the address value
- A 2-bit register (stype) to store the source address mode (frozen, incremented, or decremented)
- A 2-bit register (ssize) to store the source target data path size (byte, half-word, or word)

Reading the register with the ldf instruction has no side effects and gives the address value of the next data that will be read by the SDMA during an ldf MD instruction.

Writing the source address register may have side effects. If there is valid write data in the data register and the source address is changed, the write data is discarded. If the prefetch bit is set, a DMA read cycle is issued with the new address.

When PSA is to be written, you must specify the source target address mode, providing its size (byte, half-word, or word). This enables omission of the size field in all ldf MD instructions. When DMA performs a read cycle, its size is given by the value of the PSA source size register (ssize). If source is a memory in incremented mode, first programmed in word mode (stf PSA|SZ32I), and if an SDMA script needs to read bytes from this memory, the size of the source target must be updated before executing new accesses. The source address mode and its size are given by labels added to the stf PSA instruction as described in the write section. The ssize and stype registers are part of the DMA status register (PS).

Writing to PSA may issue an immediate error if the source size is not compatible with the value to be written into the PSA register. For instance, writing a 2 in PSA and specifying that it is memory-accessed in word mode creates an immediate error.

#### **41.4.12.2.2 Peripheral Destination Address Register (PDA)**

The destination address register contains a pointer to a source peripheral or a memory associated with the next write data transfer. It has byte granularity.

It is based on the following:

- A 32-bit register (PDA) to store the address value
- A 2-bit register (dtype) to store the destination address mode (frozen, incremented, or decremented)
- A 2-bit register (dsize) to store the destination target data path size (byte, half-word, or word)

Reading the register with the ldf instruction has no side effects, and gives the address value of the next data that will be written by SDMA during an stfMD instruction. Writing the destination register has no side effect. Similar to the PSA register, the destination address mode and source are specified in the stf PDA instruction and may also generate an error in case of incorrect programming.

#### 41.4.12.2.3 Peripheral Data Register (PD)

The data register of the peripheral DMA is a 32-bit register. When the destination address is correctly set up, any writing to PD will automatically flush the new input data.

The number of SDMA bytes that will be transferred is given by the PDA size register. Unlike other SDMA DMAs, PD is not a FIFO: It is not used to accumulate bytes that from the SDMA and must be packed before being sent to external memories. In read mode, and if the source address is correctly set up, an ldf instruction will empty PD. If a prefetch is required along with the instruction, the DMA will initiate a new read transfer.

Reading PD in prefetch mode only stalls the SDMA when the prefetched data is not yet available. Writing PD only stalls the SDMA if the previous write operation was not completed. As soon as the previous operation is over, the acknowledge is sent back to the SDMA RISC engine.

An error flag-part of PS-is set when an external access fails. The error is thus reported to the next SDMA instruction that involves the peripheral DMA.

#### 41.4.12.2.4 Peripheral State Register (PS)

The state register contains the DMA state-machine value. It can be accessed in case of an error received during a transfer.

Although all PS fields can be written by an stf instruction, it is recommended to access only the error bit (to reset it). Modifying other PS fields will provide an un-guaranteed DMA behavior.

The initialization value of PS is 0, and it consists of the following structure:

**Table 41-28. PS Structure**

	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	0	0	0	0	0	0	0	ssize		stype		dsize		dtype	
W																
R	0	0	0	0	0	d	e		0	0	0	0	0	n		
W																

**Table 41-29. PS Field Descriptions**

Field	Description
31-24	Reserved
23-22 ssize	Source Target Size. Determines the size of the read transfers on the external bus. It should match the accessed device characteristics.  00 <i>reserved</i> 01 Byte (8 bits) 10 half-word (16 bits) 11 word (32 bits)
21-20 stype	Source address Mode. Determines whether PSA is incremented, decremented, or kept unmodified after every read from the external bus.  00 Frozen Mode 01 Incremented Mode 10 Decrement Mode 11 <i>reserved</i>
19-18 dsize	Destination Target Size. Determines the size of the write transfers on the external bus. It should match the accessed device characteristics.  00 <i>reserved</i> 01 Byte (8 bits) 10 half-word (16 bits) 11 word (32 bits)
17-16 dtype	Destination address Mode. Determines whether PDA is incremented, decremented, or kept unmodified after every write on the external bus.  00 Frozen Mode 01 Incremented Mode 10 Decrement Mode 11 <i>reserved</i>
15-11	Reserved

Table continues on the next page...

**Table 41-29. PS Field Descriptions (continued)**

Field	Description
10 d	Direction Flag or DMA Mode. DMA is in write mode when data was written into PD by stf PD instructions, or if a previous DMA cycle on the external bus was a write access. Writing PDA or PSA does not change the DMA mode.  DMA is in read mode when a previous DMA cycle was a read access, and DMA stays in read mode when data is read by the SDMA with an ldf PD instruction. Reading PDA or PSA does not change the DMA mode.  0 Read Mode 1 Write Mode
9-8 e	Error. Indicates if the previous access was acknowledged with a bus error.  00 No error was received.  01 <i>reserved</i>  10 Error mode  11 Error read
7-3	Reserved
2-0 n	number of bytes in PD

**NOTE**

dtype, dsize, stype, and ssize are updated when PSA and PDA are written.

**41.4.12.2.5 Peripheral DMA Write (stf)-Write Mode**

When written by an stf instruction, the function code bits are interpreted as follows:

**Table 41-30. STF Code Bits**

Register	7	6	5	4	3	2	1	0
PSA	s		p	ar	am		sz	
PDA								
PD			pdsel					
PS			pssel					

**Table 41-31. STF Code Bits Field Descriptions**

Field	Description
7-6 s	Functional Unit selector  11 for Peripheral DMA

*Table continues on the next page...*

**Table 41-31. STF Code Bits Field Descriptions (continued)**

Field	Description
5 p (PSA)	Prefetch Flag 0 no prefetch 1 automatic prefetch
4 ar (PSA/PDA)	Address Register Selector 0 PSA 1 PDA
3-2 am (PSA/PDA)	Address Mode. Determines how PSA or PDA is modified after every read or write access to the PD. 00 Frozen-Address registers are not modified after the transfer. 01 Incremented-Address registers are incremented by the number of transferred bytes. 10 Decrement-Address registers are decremented by the number of transferred bytes. 11 Updated-PSA and PDA are not modified. Either address mode is not modified, but the width of the data path is updated by the sz field.
1-0 sz	Transfer Size 00 <i>reserved</i> 01 byte (8 bits) 10 half-word (16 bits) 11 word (32 bits)
5-0 pdsel	PD access selector 001000 is the only valid option
5-0 pssel	PS access selector 111111 writes to PS 001100 only clears the error flag in PS

Due to the large number of possible stf instructions, the following table provides only a short list of all the possible write instructions:

**Table 41-32. Peripheral DMA STF Instruction List**

Binary	Assembly	Comments
11_00_00_01 11_00_00_10 11_00_00_11	stf Rn, PSAISZ8 IF stf Rn, PSAISZ16IF stf Rn, PSAISZ32IF	<ul style="list-style-type: none"> <li>Source is a byte, half-word, or word target at the Rn address. Any further PD read instructions will trigger a byte, half-word, or word access to the source.</li> <li>Source address is frozen.</li> </ul>
11_10_00_01 11_10_00_10 11_10_00_11	stf Rn,PSAISZ8 IFIPF stf Rn,PSA ISZ16IFIPF stf Rn,PSA ISZ32IFIPF	<ul style="list-style-type: none"> <li>Source is a byte, half-word, or word target at the Rn address. Any further PD read instructions will trigger a byte, half-word, or word access to the source.</li> <li>1, 2, or 4 bytes are <i>fetched</i> from the peripheral source.</li> <li>Source address is frozen.</li> </ul>

*Table continues on the next page...*



**Table 41-32. Peripheral DMA STF Instruction List (continued)**

Binary	Assembly	Comments
11_00_01_01 11_00_01_10 11_00_01_11	stf Rn, PSAISZ8 II stf Rn, PSAISZ16II stf Rn, PSAI SZ32II	<ul style="list-style-type: none"> <li>Source is a byte, half-word, or word target at the Rn address. Any further PD read instructions will trigger a byte, half-word, or word access to the source.</li> <li>Source address is in incremented mode: <math>PSA = PSA + 1, 2</math> or 4 after read PD.</li> </ul>
11_10_01_01 11_10_01_10 11_10_01_11	stf Rn, PSAISZ8 IIIPF stf Rn, PSAISZ16IIIPF stf Rn, PSAISZ32IIIPF	<ul style="list-style-type: none"> <li>Source is a byte, half-word, or word target at the Rn address. Any further PD read instructions will trigger a byte, half-word, or word access to the source.</li> <li>Source address is in incremented mode: <math>PSA = PSA + 1, 2</math>, or 4 after read PD.</li> <li>1, 2, or 4 bytes are <i>fetched</i> from the peripheral source.</li> </ul>
11_00_10_01 11_00_10_10 11_00_10_11	stf Rn, PSAISZ8 ID stf Rn, PSAISZ16ID stf Rn, PSAISZ32ID	<ul style="list-style-type: none"> <li>Source is a byte, half-word, or word target at the Rn address. Any further PD read instructions will trigger a byte, half-word, or word access to the source.</li> <li>Source address is in incremented mode: <math>PSA = PSA - 1, 2</math>, or 4 after read PD.</li> </ul>
11_10_10_01 11_10_10_10 11_10_10_11	stf Rn, PSAISZ8 IDIPF stf Rn, PSAISZ16IDIPF stf Rn, PSAISZ32IDIPF	<ul style="list-style-type: none"> <li>Source is a byte, half-word, or word target at the Rn address. Any further PD read instructions will trigger a byte, half-word, or word access to the source.</li> <li>Source address is in incremented mode: <math>PSA = PSA - 1, 2</math>, or 4 after read PD.</li> <li>1, 2, or 4 bytes are <i>fetched</i> from the peripheral source.</li> </ul>
11_00_11_01 11_00_11_10 11_00_11_11	stf Rn, PSAISZ8 IU stf Rn, PSAISZ16 IU stf Rn, PSAI SZ32 IU	<ul style="list-style-type: none"> <li><i>Update</i> source pointer to memory, which becomes a pointer to a memory accessed in byte, half-word, or word.</li> <li>PSA value is not modified by Rn.</li> <li>Bytes present in PD are lost.</li> </ul>
11_10_11_01 11_10_11_10 11_10_11_11	stf Rn, PSAISZ8 IPFIU stf Rn, PSAISZ16 IPFIU stf Rn, PSAISZ32 IPFIU	<ul style="list-style-type: none"> <li><i>Update</i> source pointer, which becomes a pointer to a target accessed in byte, half-word, or word.</li> <li>PSA value is not modified by Rn.</li> <li>Bytes present in PD are lost.</li> <li>1, 2, or 4 bytes are <i>fetched</i> from the memory source.</li> </ul>
11_01_00_01 11_01_00_10 11_01_00_11	stf Rn, PDAISZ8 IF stf Rn, PDAISZ16IF stf Rn, PDAISZ32IF	<ul style="list-style-type: none"> <li>Destination is a byte, half-word, or word target at the Rn address, and any further PD write instructions will trigger byte, half-word, or word access to the destination.</li> <li>Destination address is frozen.</li> </ul>
11_01_01_01 11_01_01_10 11_01_01_11	stf Rn, PDAISZ8 II stf Rn, PDAISZ16II stf Rn, PDAI SZ32II	<ul style="list-style-type: none"> <li>Destination is a byte, half-word, or word target at the Rn address, and any further PD write instructions will trigger byte, half-word, or word access to the destination.</li> <li>Destination address is in incremented mode: <math>PDA = PDA + 1, 2</math>, or 4 after write PD.</li> </ul>
11_01_10_01 11_01_10_10 11_01_10_11	stf Rn, PDAISZ8 ID stf Rn, PDAISZ16ID stf Rn, PDAISZ32ID	<ul style="list-style-type: none"> <li>Destination is a byte, half-word, or word target at the Rn address, and any further PD write instructions will trigger byte, half-word, or word access to the destination.</li> <li>Destination address is in incremented mode: <math>PDA = PDA - 1, 2</math>, or 4 after write PD.</li> </ul>
11_01_11_01 11_01_11_10 11_01_11_11	stf Rn, PDAISZ8 IU stf Rn, PDAISZ16 IU stf Rn, PDAI SZ32 IU	<ul style="list-style-type: none"> <li><i>Update</i> destination pointer to memory, which becomes a pointer to a memory accessed in byte, half-word, or word.</li> <li>PDA value is not modified by Rn</li> <li>bytes present in PD are lost</li> </ul>
11_00_10_00	stf Rn, PD	<ul style="list-style-type: none"> <li>Write "dsize" bytes of Rn in PD and automatically flush to destination target</li> </ul>

Table continues on the next page...

**Table 41-32. Peripheral DMA STF Instruction List (continued)**

Binary	Assembly	Comments
11_11_11_11	stf Rn, PS	• Write status register
11_00_11_00	stf Rn, clrefPS	• Clear error flag if set

**NOTE**

When writing PD, size information is not important: It is embedded in the dsize field of PDA register. If dsize is 1, 2, or 4, then one, two, or four bytes from Rn is written to the PD register, and automatically flushed out to the destination target.

**41.4.12.2.6 Peripheral DMA Read (ldf)-Read Mode**

When received from an ldf instruction, the function code bits are interpreted as follows.

**Table 41-33. LDF Code Bits**

Register	7	6	5	4	3	2	1	0
PSA	s			ar	a			
PDA								
PD			p	cpy				
PS			pssel					

**Table 41-34. LDF Code Bits Descriptions**

Field	Description
7-6 s	Functional Unit selector 11 for Peripheral DMA
5 p (PD)	Prefetch Flag 0 no prefetch 1 automatic prefetch
4 ar (PSA/PDA)	Address Register Selector 0 PSA 1 PDA
4 cpy (PD)	Copy Mode 0 standard access 1 copy mode access
3 a	Register Set selection 0 PSA or PDA 1 PD or PS
5-0 pssel	PS access selector 111111 is the only valid option to read PS

**Table 41-35. Peripheral DMA LDF Instruction List**

Binary	Assembly	Comments
11_0_0_0_000	ldf Rn, PSA	Reads 32-bit of PSA value
11_0_1_0_000	ldf Rn, PDA	Reads 32-bit of PDA value
11_0_0_1_000	ldf Rn, PD	Reads programmed source size bytes of PD (0-extended)
11_1_0_1_000	ldf Rn, PDIPF	Reads programmed source size bytes of PD (0-extended), and starts a prefetch at PSA address.
11_0_1_1_000	ldf Rn, PDICOPY	Starts a copy transfer from the source target at the PSA address to the destination target at the PDA address. No data transmits through Rn, but Rn contents are lost (Rn is loaded with PD temporary contents that are <i>not</i> the copied data).
11_111111	ldf Rn, PS	Reads 32-bit of PS value

**NOTE**

When reading PD, size information is not important: It is embedded in the ssize field of the PSA register. If ssize is 1, 2, or 4, the one, two, or four bytes is transferred from PD to Rn. Read data is 0-extended.

**41.4.12.2.7 Peripheral DMA Unit Copy Mode**

Like burst DMA, the peripheral DMA unit has a copy mode that is used when data transfers do not involve SDMA general registers.

Data is read from the source target at a PSA address, stored in PD, and then automatically flushed to the destination target at the PDA address. Copy mode is only available for transfers that involve two targets of the same data path width.

Since copy mode is invoked with an ldf instruction, the *loaded* general purpose register loses its previous contents. (However, the new contents are unpredictable as they depend on temporary values that are seen on the external DMA bus.)

**41.4.12.2.8 Error Management**

Peripheral DMA generates two kinds of errors: the immediate error that sanctioned incorrect register programming; and the error triggered by the previous access and stored in the error flag of PS until a DMA instruction is executed.

**41.4.12.2.8.1 Immediate Errors**

The following table lists all incorrect DMA register setups.

**Table 41-36. Immediate Errors with Peripheral DMA**

Rn[1:0] values	DMA instruction	Comments
0x01 0x11	stf Rn, PSAISZ16IF stf Rn, PSAISZ16II stf Rn, PDAISZ16IF stf Rn, PDAISZ16II	If PSA points to a half-word peripheral or to a half-word address in memory, its value must be 0 modulo 2.
0x01 0x10 0x11	stf Rn, PSAISZ32IF stf Rn, PSAISZ32II stf Rn, PDAISZ32IF stf Rn, PDAISZ32II	If PSA points to a word peripheral or to a word address in memory, its value must be 0 modulo 4.
PSA[1:0]-PDA[1:0]	DMA instruction	Comments
0x01 0x10 0x11	stf Rn, PSAISZ32IU stf Rn, PDAISZ32IU	When PDA or PSA is updated and becomes a pointer to a word address in memory, its content must be 0 modulo 4.
0x01 0x11	stf Rn, PSAISZ16IU stf Rn, PDAISZ16IU	When PDA or PSA is updated and becomes a pointer to a half-word address in memory, its content must be 0 modulo 2.
Read/Write PD instruction	Comments	
stf Rn,PD ldf Rn,PD	If PDA size (dsize) has never been set up before an stf PD instruction (dsize=0) If PSA size (ssize) has never been set up before an ldf PD instruction (ssize=0)	
ldf Rn,PDICPY	Copy mode is possible only between two targets whose data path width is identical.  It is P8↔P8, P16↔P16, or P32↔P32 regardless of the way the address registers are incremented.	

#### 41.4.12.2.8.2 Data Transfer Errors

When PSA and PDA are correctly set up, the only error that may arise for an ldf PD or stf PD instruction would be the error of the previous DMA cycle.

Error handling is driven by a single consideration: When an error occurred during a data read on the DMA interface, this error should appear as a transfer error to the core when the core attempts to retrieve the data that was not successfully read from the accessed device (memory or peripheral).

When an error occurred during a write access to the DMA interface, the data is still available in PD and should not be destroyed by subsequent core accesses: The core must be warned about the error issue.

There are three error handling mechanisms for each case: [Read Error \(First Phase\)](#), [Write Error and Read Error \(Second Phase\)](#), and [Copy Mode Errors](#) handling.

#### 41.4.12.2.8.3 Read Error (First Phase)

If an error occurred during a prefetch command, the peripheral DMA enters its ERROR READ mode (PS[9:8]=11). In this mode, the error is reported on the next ldf PD instruction and writing PSA, PDA, or PD will cancel the error flag.

The block returns no error mode and instructions are normally executed (a DMA cycle may be triggered). Similarly, initiating a copy transfer will reset the error flag and start a copy transfer. The following table details which instructions can be executed in this mode.

**Table 41-37. Possibilities in ERROR READ Mode**

DMA Instruction	Immediate Error	Comments
stf rn, PD stf rn, PSA (IU IPF) stf rn, PDA ldf rn, PDICOPY	NO	Error mode is reset, PSA or PDA are updated, or a write cycle is started. For the ldf PDICOPY, a copy loop is executed.
stf rn, PS	NO	PS is updated.
ldf rn, PS ldf rn, PSA ldf rn, PDA	NO	PS, PSA, and PDA could be read in ERROR READ mode without any side effects (for example, no DMA cycle is triggered).
ldf rn, PD	YES	Error of the previous read access is reported here and the peripheral DMA enters its ERROR mode.

#### 41.4.12.2.8.4 Write Error and Read Error (Second Phase)

The peripheral DMA enters its ERROR mode (PS[9:8]=10) when the previous DMA write cycle failed, or, as explained in [Read Error \(First Phase\)](#), when an ldf PD is executed while the block is in ERROR READ mode. When a DMA cycle failed, address registers (PSA, PDA) are not modified and continue to point to the problematic address. In ERROR mode, stf instructions may raise an immediate error, and ldf instructions will not (as detailed in the table below).

**Table 41-38. Possibilities in ERROR Mode**

DMA Instruction	Immediate Error	Comments
stf rn, PD stf rn, PSA stf rn, PDA	YES	Any attempt to modify PD, PSA, or PDA will raise an immediate error, and the peripheral DMA stays in ERROR mode. When address registers are write accessed, an error is returned.
stf rn, PS	NO	This is the only way to exit the ERROR mode. PS[3] must be reset by an stf PS instruction.
ldf rn, PS ldf rn, PSA ldf rn, PDA	NO	PS, PSA, and PDA could be read in ERROR mode without any side effects (for example, no DMA cycle is triggered).
ldf rn, PD	YES	Whatever the DMA direction (read or write), an ldf rn, PD instruction will show an immediate error.

#### 41.4.12.2.8.5 Copy Mode Errors

Because copy mode is a write access that follows a read access, there are two possible cases of bus error.

When the read access incurs a bus error, the peripheral DMA behaves exactly as described in [Read Error \(First Phase\)](#) and [Write Error and Read Error \(Second Phase\)](#) : It enters its ERROR READ mode, and so on.

When the error occurred during the write access of the copy transfer, the DMA enables the core to retrieve the data that was read because it is assumed the read from the peripheral removed the data from its source device. Therefore, the data to be flushed is still in PD. Any subsequent access to PD triggers an error to the core, which should execute its error handling procedure.

Once the ERROR mode is left (after writing to PS), it is possible for the core to retrieve the data in PD with an ldf instruction or try to flush PD contents once again (for example, when the error was due to a full FIFO and the script waited for the FIFO to be emptied) with another ldf instruction in copy mode. This latter instruction detects that there is valid data in PD, tries to flush it, and thus skips the read phase of the copy instruction. This is a different behavior from the usual stf PD instruction that overwrites PD with the selected General Purpose register contents. The same mechanism can be used any time PD holds data that is not written because of a bus error on the DMA interface; when the data was written via a copy instruction, or via the usual stf PD instruction.

#### 41.4.12.2.8.6 Error Check Example

The following code illustrates an example checking for both immediate and data transfer errors on a store to the PD register. The first bdf instruction checks for an immediate error, but if a data transfer error occurred it is reported until the next instruction to access the Peripheral DMA. A second check of the error flags is done after the ldf PS instruction. The value of PS here can be ignored. The act of reading any register in Peripheral DMA while it is in an error mode that returns the error to the core to set either the SF or DF flag. Any error returned on an ldf command sets the SF flag and any error returned on an stf instruction sets the DF flag. This can create a situation as shown in the example where a bus error during a DMA write which would normally be considered as a destination fault is reported as a source fault because the error was reported to the SDMA core during an ldf instruction.

##### Peripheral DMA Error Check

```

    clrf    0           // Clear SF and DF flags
    stf     R4, PD       // Write data to memory
    bdf     error_routine // Check for immediate error from write to PD.
    ldf     r3, PS       // Read PS (PS value in R3 can be ignored)

```

```
bsf    error_routine    // Check for bus error from "stf R4,PD"
        // SF is set because it is a ldf instruction, even though
        // the original error was a destination fault
```

#### 41.4.12.2.9 Peripheral DMA Unit Prefetch/Flush Management

There is no flush bit because every time data is stored in PD by a stf PD instruction—assuming PDA is correctly programmed—it is automatically flushed to the destination.

An acknowledge is returned in the cycle of the DMA instruction, and the SDMA is only stalled by an instruction that addresses the peripheral DMA when the previous DMA access is not over.

#### 41.4.12.3 OnCE and Real-Time Debug

The On-Chip Emulation block (OnCE) is the debug interface to the SDMA.

It supports the access to all core internal devices (registers, memory, and so on), and provides a set of mechanisms that control the core. The OnCE is accessed by JTAG ports at the chip's board level, or by the host via its peripheral bus.

To reduce the size of the hardware material involved, all tasks supported by the OnCE are performed on the SDMA core. The architecture of the SDMA OnCE is relatively simple and very flexible.

The commands supported by the SDMA OnCE are listed in the following sections.

##### 41.4.12.3.1 Memory and Register Access

A set of mechanisms is provided to access SDMA memory and register locations. Both reading and writing are allowed. The access is supported if the processor is in debug mode.

Those registers can also be accessed through the ARM platform Control interface when the OnCE is controlled by the ARM platform, as described in the "Using BP" section.

##### 41.4.12.3.2 Hardware Breakpoints

An event detection unit is implemented to support memory breakpoints. The unit watches the data exchanged between the SDMA memory bus and the core.

A debug request is sent to the core when matching conditions occur. The unit supports mixed conditions based on address range, access type, and data value. Event detection unit configuration registers are memory mapped in the SDMA space (see [ARM platform Channel 0 Pointer \(SDMAARM\\_MC0PTR\)](#)): You can modify them through a regular memory access or the ARM platform control interface.

#### **41.4.12.3.3 Watchpoints**

One output pin is provided to monitor matching trigger conditions that are defined in the event detection unit.

#### **41.4.12.3.4 Software Breakpoints**

The SDMA instruction set contains a software breakpoint. Upon executing a software breakpoint instruction, the core suspends normal execution and enters debug mode.

No hardware step execution mode is implemented in the OnCE, but this feature may be implemented at the software level with this instruction.

#### **41.4.12.3.5 Core Control**

Commands are provided to monitor and control processor activity. You can halt the core, rerun the core from another address location, and get processor status.

Any hardware breakpoint on the instruction bus is not supported, but this feature may be implemented by inserting a software breakpoints program.

### **41.4.13 The OnCE Controller**

The OnCE controller receives commands from the ARM platform or from the JTAG controller. Each command is interpreted before being sent to the core.

#### **41.4.13.1 OnCE Commands**

A small set of commands supports the communication between the OnCE and the external world.

This command set enables you to perform any of the following tasks: control processor activity, save core context, and execute an SDMA instruction from the OnCE. Combined together, these tasks perform more complex commands.



A full OnCE command contains a 4-bit instruction (the OnCE command opcode) and a variable length data field (the OnCE data). During command execution, the OnCE data is transferred in a OnCE internal register before being exchanged with the SDMA. Some data values are also exported. This mechanism creates a link between the processor and the external world. Nine commands are defined: The following table presents their formats.

**Table 41-39. OnCE Command Opcode Values**

Instruction Opcode	Name	Action	Register	Data Field Size	Mode
0000	rstatus	Reads the OnCE status register	STATUS	16-bit	normal/debug
0001	dmov	Updates general register GReg1	GREG1	32-bit	debug
0010	exec_once	Runs the instruction from the SDMA instruction register	INSTRUCTION	16-bit	debug
0011	run_core	Returns to normal execution	BYPASS	1-bit	debug
0100	exec_core	Returns to normal execution via a jump instruction that specifies the new address	INSTRUCTION	16-bit	debug
0101	debug_rqst	Stops the core after execution of current instruction	BYPASS	1-bit	normal
0110	rbuffer	Reads the real time buffer	RTB	32-bit	normal/debug
0111-1110	reserved	Reserved	BYPASS	1-bit	normal/debug
1111	bypass	Bypasses TARM platform controller	BYPASS	1-bit	normal/debug

Each instruction corresponds to a specific action performed on the OnCE. The nature of the associated data field is clearly identified. The dmov command is followed by a 32-bit data value (which is a data value for the SDMA); the exec\_once and the exec\_core commands are followed by a 16-bit data value (which is an instruction for the SDMA); the rstatus command is followed by a 16-bit control value (which is the content of the OnCE status register); the rbuffer command is followed by a 32-bit data value. The debug\_rqst and the run\_core commands are followed by a single bit data field (this is a bypass value). Finally, the bypass instruction enables the SDMA JTAG TAP controller to be daisy-chained with another JTAG TAP controller. This is a JTAG-only feature. The set of commands is simple, but enables you to perform any possible task on the SDMA during a debug process.

#### 41.4.13.2 Sending Commands to the OnCE Controller

The JTAG access is the standard access to the OnCE, but sometimes the JTAG is not available to fix some bugs (if the chip is in production for instance), an additional access is then required. Therefore, one ARM platform access to the OnCE is provided.

### 41.4.13.2.1 Using the JTAG Interface

A serial access is performed through the five JTAG pins TCK, TRST, TMS, TDI, and TDO. A Test Access Port controller is provided to decode the TMS control signal.

It produces shift-enable signals (shift\_ir and shift\_dr), and updates enable signals (update\_ir and update\_dr). It is fully compliant with the IEEE 1149.1 testability (JTAG) standard.

During the shift\_ir state, the command opcode is shifted into the OnCE controller (for example, the signal from the TDI pin is shifted into the command register and the TDO pin receives the signal shifted out). After transferring the four bits of the command, an update\_ir signal is asserted and the command is decoded. The target data register is now clearly identified and the corresponding control signal is produced, as follows: bypass enable signal (bp\_en), instruction enable signal (inst\_en), data enable (data\_en), and status enable signal (stat\_en).

During the shift\_dr state, the TDI signal is shifted into one of the following target registers: bypass register (1 bit), SDMA instruction register (16 bits), SDMA data register (32 bits), or OnCE status register (16 bits). The TDO pin is connected to the output of the selected register to receive the signals shifted out.

The JTAG access is disabled when the ARM platform access is enabled.

### 41.4.13.2.2 Using the ARM platform

The ARM platform access to the OnCE is not the standard access, but it is required if the JTAG is not available.

For example, if the SDMA ROM is out of use on a chip in production, and the ARM platform needs to download new code and restart the SDMA, the OnCE can easily perform this operation. This type of debug operation justifies the use of an ARM platform access to the OnCE.

To drive the OnCE, the ARM platform uses some registers contained in the ARM platform Control block of the SDMA. These registers are accessed through the ARM platform peripheral bus. Most of these registers are connected to another register in the OnCE controller. Thus, accessing one of these registers is equivalent to accessing the associated register in the OnCE controller.

The set of registers in the ARM platform Control block is listed below:

- **ONCE\_ENB** register (1 bit, read/write)-This 1-bit register enables the ARM platform access to the OnCE. When this bit is set, the signals from the JTAG are ignored.

When it is cleared, all writing operations to the following registers through the Host Control interface are ignored. This register is reset on a JTAG reset.

- **ONCE\_CMD** register (4 bits, read/write)-This 4-bit register receives the command opcode. It is connected to the command register in the controller. A write access to this register causes the associated command to be executed on the OnCE. For example, after writing "0001" in this register, a dmov command is executed.

### NOTE

On the ARM platform side, the rstatus and bypass commands are not supported. This register is reset on a JTAG reset.

- **ONCE\_DATA** register (32 bits, read/write)-This 32-bit register is connected to the SDMA data register. This register is used when executing a dmov or rbuffer command.

### NOTE

Before requesting a dmov command, the 32-bit data to transfer must be written in the ONCE\_DATA register. At the end of the execution, the register is updated with GReg1 former value. This register is reset on a JTAG reset.

- **ONCE\_INSTR** register (16 bits, read/write)-This 16-bit register is connected to the SDMA instruction register. This register is used when executing an exec\_core or an exec\_once command.

### NOTE

Before requesting an exec\_core or an exec\_once command, the appropriate instruction must be written in the ONCE\_INSTR register. This register is reset on a JTAG reset.

- **ONCE\_STAT** register (16 bits, read only)-A read access to the ONCE\_STAT register returns the content of the OnCE status register (OSTAT). This register is read only.
- The bypass register is not useful when the ARM platform controls the OnCE, therefore no register is defined in the ARM platform Control block to access the bypass register.

#### 41.4.13.2.3 Conflicts Between the JTAG and the ARM platform Accesses

When ARM platform access to the SDMA OnCE is enabled (that is, when the bit in the ONCE\_ENB register is set), the JTAG access is disabled. This guarantees that the block is not accessed at the same time on both sides.

It is possible to check whether the JTAG access to the SDMA OnCE is enabled from the JTAG port. When the JTAG access is disabled, the SDMA TDO always returns 1. The check requires the following steps:

- Execute a dmov command from debug mode (with neither 0xffffffff nor 0x0 as dmov value: 0x5a5a5a5a is good).
- Execute another dmov command (the value here is not important).
- The returned value from the latter dmov command should be the original one if the JTAG access is enabled; if it is 0xffffffff instead of the original input value, this means the JTAG access is disabled.

### **41.4.13.3 Executing a Command from the OnCE**

All the commands defined in [OnCE Commands](#) can be accessed through the JTAG. The ARM platform can access all these commands except the rstatus command.

On the ARM platform side, the OnCE status is directly accessed by reading the ONCE\_STAT register.

#### **41.4.13.3.1 Nature of the Commands**

Two types of commands may be distinguished. First, there are two commands that do not interact with the core: rstatus and rbuffer. Those commands may be requested at any time: They do not depend on the core status.

#### **NOTE**

Each of these commands exports a data value or a status value from the SDMA.

There are also commands that interact with the core: dmov, run\_core, exec\_core, exec\_once, and debug\_rqst. These commands are core status dependent, as follows:

- During user mode only the debug\_rqst is taken into account.
- During debug mode, all these commands are taken into account except the debug\_rqst. For example, an exec\_once command requested while not in debug mode has no effect.

#### **41.4.13.3.2 Execution Request**

The SDMA starts executing a task in debug mode when requested by the OnCE controller. The execution starting time depends on the type of access used to communicate with the OnCE.

If the JTAG is used, the request is sent after decoding the `update_dr` state in the TAP controller. Therefore, always cross this state when sending a command through the JTAG. If the OnCE is driven from the ARM platform side, the request is sent after detecting a write access to the `ONCE_CMD` register. All the registers involved in this operation must be loaded first.

The following is an example of an `exec_core` command execution from the ARM platform side: After writing '010' in the `ONCE_CMD` register, the OnCE controller asks the SDMA to execute the instruction contained in the `ONCE_INSTR` register. The instruction involved should be available in the `ONCE_INSTR` register before the beginning of the execution.

#### 41.4.13.3.3 Command Execution

The following list shows the commands and details how each command is executed:

- **rstatus command execution**-The `rstatus` command exports the content of the OnCE status register (OSR). If the JTAG is used, the status information is captured in the OnCE status register during the `capture_dr` state, and shifted out after 16 TCK clock cycles in the `shift_dr` state. The `rstatus` command is not supported on the ARM platform side, but a status register is provided instead. The `rstatus` may be performed in both debug and user modes.
- **dmov command execution**-The `dmov` command accesses SDMA internal registers. Executing a `dmov` instruction exchanges the 32-bit data values between the SDMA data register and the general register `GReg1`.
- If the JTAG is used, the content of `GReg1` is captured in the SDMA data register during the `capture_dr` state, then it is shifted out after 32 TCK clock cycles in the `shift_dr` state. During the `update_dr` state, `GReg1` is updated with the new, shifted-in 32-bit data value. If the OnCE is driven from the ARM platform side, the data values contained in `GReg1` and the SDMA data register are exchanged after detecting a write access to the `ONCE_CMD` register. The `ONCE_DATA` register must therefore be loaded first.
- **exec\_once command execution**-The `exec_once` command executes the instruction loaded in the SDMA instruction register. The command may only be requested from debug mode. The SDMA returns to debug mode at the end of the execution.
- **Change of flow instructions as well as instructions that may cause a context switch are not supported:** The comprehensive list comprises `done/yield/yiedge` (except `done 5`), `BF`, `BT`, `BSF`, `BDF`, `JMP`, `JSR`, `JMPR`, `JSRR`, `RET`, and `LOOP`, as well as all the illegal instructions.

No other command should be requested before the SDMA returns to debug mode. The SDMA status (for example, whether it is in debug mode or not) can be detected by polling with the rstatus OnCE command, monitoring the debug\_mode pin, or checking the [OnCE Status Register \(SDMAARM\\_ONCE\\_STAT\)](#) register via the ARM platform control interface.

### NOTE

Most of the instructions are single-cycle, which omits the step of polling the status. Loads and stores to DMA units are typical instructions that might require this polling.

If the JTAG is used, the 16-bit instruction is shifted in the SDMA instruction register after 16 TCK clock cycles in the shift\_dr state. A request is sent to the core when the update\_dr state is decoded in the TAP controller. If the OnCE is driven from the ARM platform side, the request is sent to the SDMA when detecting a write access to the ONCE\_CMD register. The ONCE\_INSTR register must be therefore be loaded first.

- run\_core command execution-The run\_core command leaves debug mode and resume normal program execution. The next instruction executed is the last instruction decoded before entering debug mode. Be sure to restore core context before re-running the core. This procedure is detailed in [Restoring the Context](#).
- If the JTAG is used, a 1-bit bypass value is shifted in the bypass register in the shift\_dr state. The SDMA is rerun when the update\_dr state is decoded in the TAP controller. If the OnCE is driven from the ARM platform side, the core is rerun when detecting a write access to the ONCE\_CMD register.
- exec\_core command execution-The exec\_core command resumes program execution from any address. The 16-bit instruction provided with the exec\_core overwrites the last instruction decoded before entering debug mode. This command is designed to support change of flow instructions, so that a program execution can be restarted from any address. After executing an exec\_core command, the SDMA leaves debug mode. The exec\_core command is usually used with a jmp instruction.
- If the JTAG is used, the 16-bit branch instruction is shifted in the SDMA instruction register after 16 TCK clock cycles in the shift\_dr state. The SDMA is rerun when the update\_dr state is decoded in the TAP controller. If the OnCE is driven from the ARM platform side, the SDMA reruns when detecting a write access to the ONCE\_CMD register. The ONCE\_INSTR register must therefore be loaded first. For example, to restart the SDMA from the program address 0x100, the instruction loaded should be a jump to address 0x100 instruction.
- debug\_rqst command execution-The debug\_rqst command puts the SDMA in debug mode. If the JTAG is used, a 1-bit bypass value is shifted in the bypass register during the shift\_dr state. A debug request is sent to the SDMA when the update\_dr

state is decoded in the TAP controller. If the OnCE is driven from the ARM platform side, the debug request is sent when detecting a write access to the ONCE\_CMD register. When the SDMA is already in debug mode, this command is simply ignored.

- **rbuffer command execution**-The rbuffer command exports the content of the real time buffer (RTB). If the JTAG is used, the content of the real time buffer (RTB) is captured in the SDMA data register during the capture\_dr state. The register is completely shifted out after maintaining the shift\_dr state during 32 TCK clock cycles. If the OnCE is driven from the ARM platform side, the content of the RTB is captured in the ONCE\_DATA register after detecting a write access to the ONCE\_CMD register.
- **bypass command execution**-This command is only available from the JTAG interface. It enables daisy-chaining of the SDMA JTAG TAP controller with other JTAG TAP controllers. This command does not change the SDMA state and can be executed in any mode (run, debug, or sleep). It selects the bypass register of the TAP controller.

#### 41.4.13.4 Registers Descriptions

See [SDMACORE](#), and [SDMAARM](#), for detailed information on each register.

##### 41.4.13.4.1 Event Cell Counter Register (ECOUNT)

The event cell counter register is a 16-bit register that contains the number of times minus one that an event detection occurs before generating a debug request.

This register should be written before attempting to use the event detection counter during an event detection process. The event cell counter register is cleared on a JTAG reset.

##### 41.4.13.4.2 Event Cell Address Registers (EAA or EAB)

The event cell contains two address registers-the event cell address register (a), called EAA, and the event cell address register (b), called EAB. Every address register is a 16-bit register that stores a user-defined address value. This value computes one of the following address conditions: addra\_cond or addrb\_cond. Every address register is cleared on a JTAG reset.

#### 41.4.13.4.3 Event Cell Address Mask Register (EAM)

The event cell address mask register is a 16-bit register that contains a user-defined address mask value. This mask is applied to the address value latched from the memory address bus before comparing addresses.

##### NOTE

There is a common address mask value for the two address comparators. If bit  $i$  of this register is set, then bit  $i$  of the address value latched from the memory bus does not influence the result of the address comparison. The event cell address mask register is cleared on a JTAG reset.

#### 41.4.13.4.4 Event Cell Data Register (ED)

The event cell data register is a 32-bit register that contains a user-defined data value. This data value is an input for the data comparator, which generates the data\_cond condition.

The event cell data register is cleared on a JTAG reset.

#### 41.4.13.4.5 Event Cell Data Mask Register (EDM)

The event cell data mask register is a 32-bit register that contains a user-defined data mask value. This mask is applied to the data value latched from the memory bus before comparing data.

Setting bit  $i$  of the event cell data mask register means that bit  $i$  of the data value latched from the address bus does not influence the result of the data comparison. The event cell data mask register is cleared on a JTAG reset.

#### 41.4.13.4.6 Real Time Buffer Register (RTB)

The real Time Buffer register is a 32-bit register that stores and retrieves run-time information without putting the SDMA in debug mode.

Refer to [Real Time Buffer](#) for more details.

#### 41.4.13.4.7 Event Control Register (ECTL)

The event cell control register is a 16-bit register that defines cell event occurrence conditions.

The event cell control register is cleared on a JTAG reset. See also [OnCE Event Detection Unit](#) for more details.



#### 41.4.13.4.8 Trace Buffer (TB)

The Trace Buffer register retrieves the information in the Trace Buffer.

See [Trace Buffer](#) for more details.

#### 41.4.13.4.9 OnCE Status Register (OSTAT)

The OnCE status register is a 16-bit register that contains processor and event detection unit status. The OSTAT is a read-only register.

Refer to [OnCE Status Register \(SDMAARM\\_ONCE\\_STAT\)](#) for detailed description of the individual fields in the OSTAT register.

The following figure shows the OSTAT structure.

**Table 41-40. OnCE Status Register (OnCE)**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PST[3:0]				RCV	EDR	ODR	SWB	MST					ECDR[2:0]		

Where PST[3:0] is the SDMA core state, RCV is set when the real-time buffer (RTB) is modified. EDR, ODR, and SWB are set, respectively, when the SDMA has entered debug mode because of an external debug request, a OnCE debug\_rqst command, or a software breakpoint. MST is set when the OnCE is controlled from the ARM platform control interface, and when ECDR is a three-flag set that shows the event cell condition(s) that put the core in debug mode. The OSTAT never provides more than one reason for entering debug mode.

There are two ways of accessing OSTAT content, as follows:

1. Send an rstatus command to the OnCE controller through the JTAG, or read the ONCE\_STAT register through the ARM platform access. Executing the rstatus command through the JTAG can be performed in both user and debug modes.
2. Perform an SDMA read access to the location in the SDMA core memory map (OSTAT register) debug mode using the exec\_once command. With this method of access, the SDMA state reflected by the PST (processor status bit) is always DATA.

The register may also be accessed by a running application.

### 41.4.13.5 JTAG Interface Requirements

Because the signals received from the JTAG (running on TCK) are transferred to the OnCE controller (running on the SDMA clock), a synchronization mechanism is required.

#### 41.4.13.5.1 TCK Speed Limitation

In the JTAG top-level layer, the TDO signal is always captured on a TCK falling edge. To guarantee a stable TDO signal from the SDMA during this operation, a falling edge detection is performed on TCK.

Before being latched in the *I* flip-flop (see [Figure 41-11](#)) on TCK falling edge, the TDO signal must be stable at the input of the flip-flop. This condition is verified if the TCK period is superior to the following delay:

*worst-case edge detection delay + negative-edge signal propagation delay + JTAG top-level logic propagation delay*

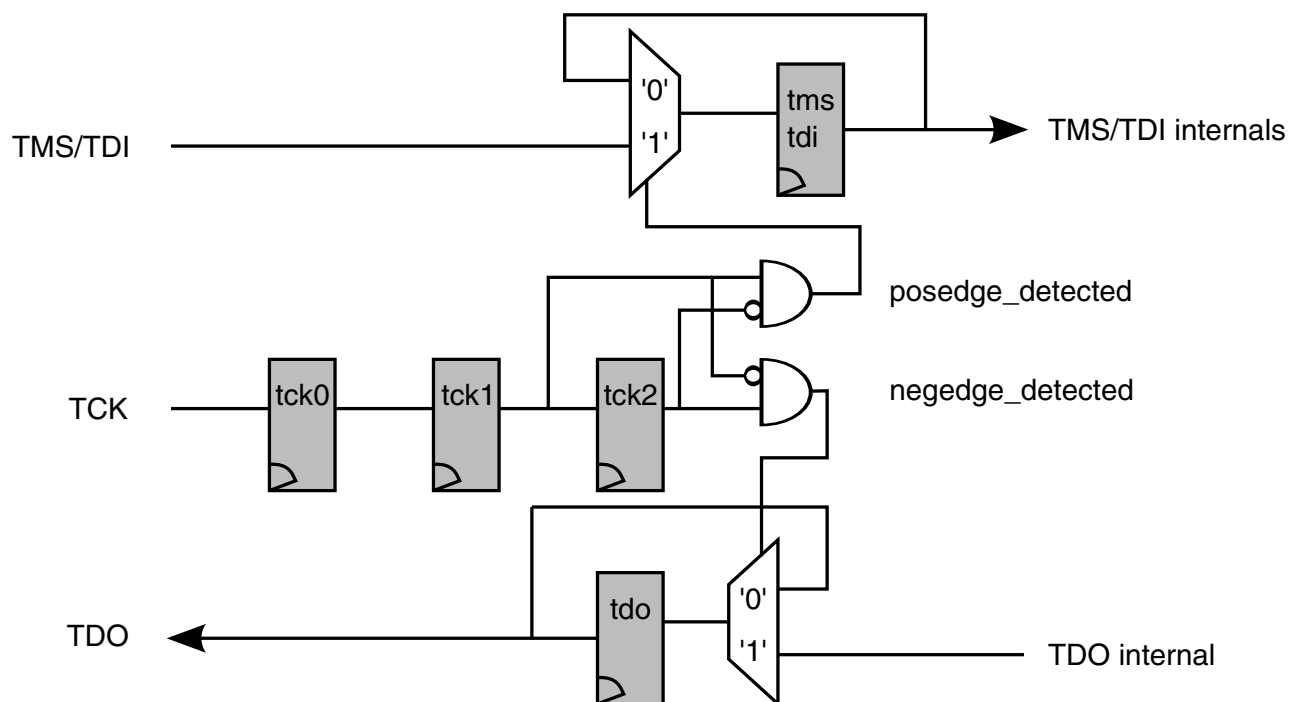
The frequency relationship,  $TCK < CLK/8$ , limitation guarantees that all operations are performed as expected.

#### 41.4.13.5.2 Synchronization Implementation

The figure found here shows the synchronization mechanism.

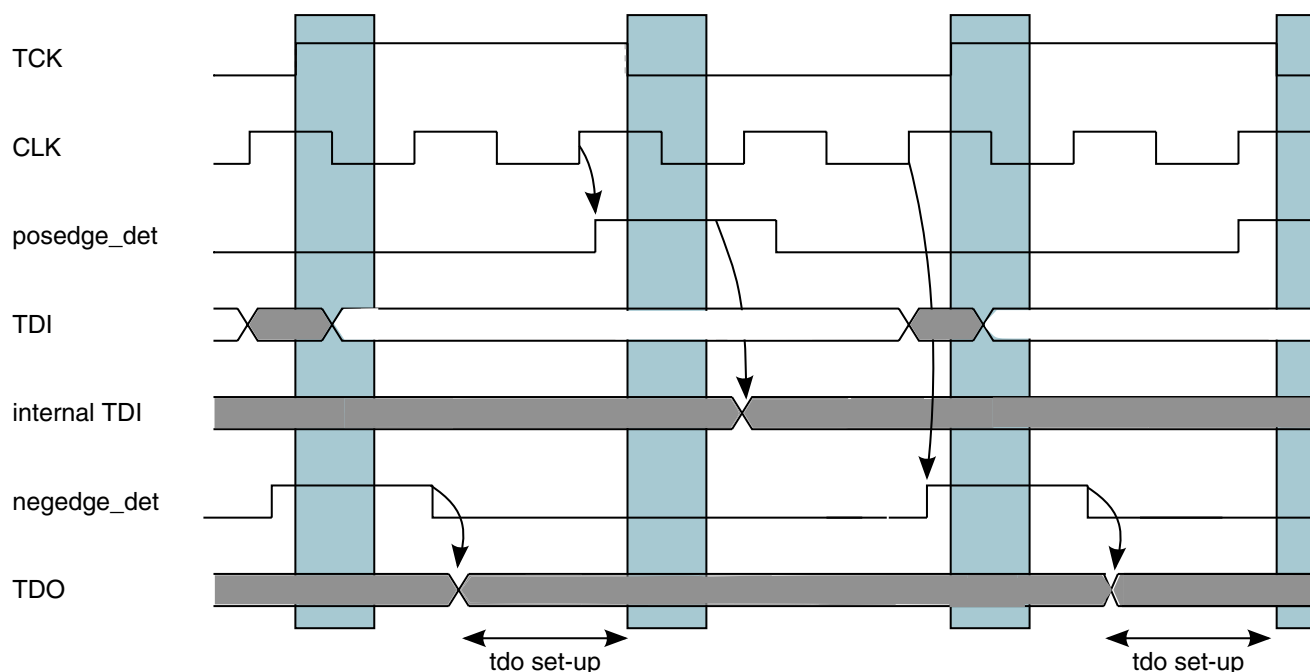
Flip-flops tck0, tck1, and tck2 perform falling- and rising-edge detections on TCK. They generate the posedge\_detected and negedge\_detected nets that are used to sample the TDI and TMS inputs into the respective tdi and tms flip-flops, and update the tdo flip-flop to yield the TDO output. In the design, the only signal that might go metastable is the output of the tck0 flip-flop. This signal is captured in the tck1 flip-flop and no logical operation is performed on it to minimize a metastability propagation risk.

The TDI and TMS flip-flops also cannot go metastable: The propagation time of the rising-edge detection signal through tck0, tck1, and tck2 guarantees that the TDI and TMS inputs are stable when captured in the TDI and TMS flip-flops.



**Figure 41-11. OnCE Synchronization Layer**

The following figure shows synchronization timings. It takes three CLK clock cycles to synchronize TDI on the SDMA clock.



**Figure 41-12. Synchronization Timings**

### 41.4.13.5.3 JTAG Controller Start-Up Recommended Procedure

To ensure correct TAP controller initialization, it is recommended to use the following procedure:

1. Assert JTAG reset TRSTB (for example, set low).
2. Set TMS low.
3. Wait for 1 TCK clock.
4. Release JTAG reset TRSTB (for example, set high).
5. Wait for a minimum of five TCK cycles.

## 41.4.14 Using the OnCE

This section provides the elements necessary to run the OnCE during a debug process.

In addition to the basic set of commands described in [OnCE Commands](#), more complex commands can be built to meet users' requirements.

### 41.4.14.1 Activating Clocks in Debug Mode

For power consumption issues, some clocks in the SDMA are disabled when not needed.

This is the case for instances when the SDMA is in sleep mode. Clock gating management depends on the interface used to control the OnCE.

- For the JTAG access, the SDMA clock gating must be turned off via the `clk_gating_off` input.
- For the ARM platform access, the SDMA clock gating is automatically turned off when the ARM platform access is enabled (see [OnCE Enable \(SDMAARM\\_ONCE\\_ENB\)](#)).

### 41.4.14.2 Getting the Current Status

Most of the commands the OnCE supports have an impact on the status of the SDMA.

It is not permissible to request the execution of an instruction on the SDMA from the OnCE while the SDMA is not in debug mode. Such a violation may cause unpredictable behavior, and it might be necessary to reset the SDMA.

Therefore, the value of the PST bits provided in the OnCE status register should always be checked before sending any request to the SDMA.

### 41.4.14.3 Methods of Entering Debug Mode

A debug request may be asserted at any time, but it is not always taken into account immediately. Debug mode cannot be entered in the middle of an instruction, or during the save or restore states of a context switch.

The request is ignored when the core is already in debug mode. Refer to [Figure 41-4](#), which shows all possible transitions to the debug state, as there are several ways to enter debug mode.

#### 41.4.14.3.1 External Debug Request During Reset

To enter debug mode after exiting reset, the external debug line has to be maintained high. This line is handled by the JTAG top-level block.

##### NOTE

The SDMA detects the debug requests only if the SDMA clock is running (see [Activating Clocks in Debug Mode](#)). The debug request line should not be maintained high when the SDMA is in debug mode.

##### NOTE

The debug\_rqst command (from the OnCE command set) is not supported during system reset.

#### 41.4.14.3.2 Debug Request During Normal Activity

During normal activity, the SDMA enters debug mode when the following is true:

1. If the debug request line from the JTAG top-level is asserted, or
2. If the OnCE controller receives a debug\_rqst command.

The debug\_rqst command can be sent by the JTAG access or by an access on the ARM platform side (if the ARM platform access is enabled).

#### 41.4.14.3.3 Software Breakpoint Instruction

The SDMA enters debug mode at the end of the execution of a software breakpoint instruction. This instruction must be inserted in program flow executed by the core.

#### 41.4.14.3.4 Event Detection Unit Matching Condition

If the event detection is enabled, a debug request is sent to the core after detecting a matching condition on the SDMA memory bus.

See [OnCE Event Detection Unit](#) for more details.

#### 41.4.14.4 Executing Instructions in Debug Mode

The OnCE supports a mechanism to execute instructions in debug mode. If the SDMA is in debug mode, then the `exec_once` command can be used to execute an SDMA instruction from the OnCE controller. The SDMA returns to debug mode at the end of each execution.

Some instructions are not supported by the `exec_once` command: `done`/`yield`/`yiedge` (except `done 5`), `BF`, `BT`, `BSF`, `BDF`, `JMP`, `JSR`, `JMPR`, `JSRR`, `RET`, and `LOOP`, as well as all the illegal instructions are not supported.

#### NOTE

While instructions are executed in debug mode from the OnCE, the program counter of the SDMA is not incremented.

#### 41.4.14.5 Command Sequences Examples

This section provides examples of command sequences that run the SDMA in debug mode. These sequences are available for both the ARM platform and JTAG accesses.

The following presents the syntax used in this section. The data field provided with each command is put in parenthesis with the command name. A '-' is used if the data field provided is a *don't care* value.

```
my_command(data_field);           // executing my_command with a data field
my_command(-);                   // executing my_command with a don't care data field
```

The value returned by the command (if there is one) is referred by an assignment. In case the value returned by the command is not used, the assignment is omitted. For an ARM platform access, the value returned (it is always a data value) is obtained by reading back into the SDMA data register.

```
data_out = my_command(data_in); // returning a data value
```

To clarify the syntax, the instructions' opcodes are referred to by their names. In practice, use the corresponding 16-bit encoding.

### 41.4.14.5.1 Getting the SDMA Status

#### NOTE

Before executing any command that affects the SDMA (like `dmov` or `exec_once`), check that the SDMA is in debug mode.

Use the following snippet:

```
rstatus();           // read SDMA status until the SDMA is in debug mode
...
rstatus();
```

If the SDMA is not in debug mode, then a debug request must be generated. In this case, the SDMA enters debug mode at the end of the execution of the current instruction. Use this snippet:

```
debug_rqst(-);      // debug request
```

In the following sections, it is assumed that the SDMA was successfully put into debug mode.

### 41.4.14.5.2 Saving the Context

The first debug task is to save the SDMA context, which is the content of the eight general-purpose registers, the loop and PC-related registers, and the flags.

Use the general register `GReg[1]` as an intermediate register to export the entire context of the SDMA.

The following example shows how to save `GReg[0]`, `GReg[1]`, `GReg[2]` and `GReg[3]`. The sequence of commands used to export additional general registers is very similar to this.

Save `GReg[0]`, `GReg[1]`, `GReg[2]`, and `GReg[3]`

```
GReg1_data = dmov(-);           // the value exported is the content of
GReg[1]                                     // puts the content of GReg[0] into
exec_once("mov GReg1,GReg0");
GReg[1]
GReg0_data = dmov(-);           // the value exported is the content of
GReg[0]                                     // puts the content of GReg[2] into
exec_once("mov GReg1, GReg2");
GReg[1]
GReg2_data = dmov(-);           // the value exported is the content of
GReg[2]                                     // puts the content of GReg[3] into
exec_once("mov GReg1, GReg3");
GReg[1]
GReg3_data = dmov(-);           // the value exported is the content of
GReg[3]
```

Get the value of the internal flags (SF, DF, T, and LM), of the loop related registers (EPC and SPC), and of the PC-related registers (PC and RPC). Use a `done 5`, which is the formatting instruction dedicated to the debug. This instruction formats the flags and the values contained in the registers. It also writes the resulting values into the channel

context memory. It should not be used when entering debug from the IDLE state (for example, with no active channel script running on the SDMA), because it will update a channel context that may belong to any channel.

```
exec_once("done 5"); // formatting the value of flags and registers
```

At this point, the channel context should be up-to-date in memory, and debug operations should now be possible. However, the context can be exported with the following instructions:

### Exporting the Context

```
dmov(ctx_base_addr); // loading GReg[1] with the channel
context base address
exec_once("ld GReg0, (GReg1,0)"); // get RPC-PC into GReg0
exec_once("ld GReg1, (GReg1,1)"); // get SPC-EPC into GReg1
Loop_data = dmov(-); // read back the value of Loop registers
exec_once("mov GReg1, GReg0"); // puts the PC info into GReg1
PC_data = dmov(-); // reads back the content of the PC registers
```

After this sequence of operations, the entire SDMA context is exported via the OnCE.

### 41.4.14.5.3 Restoring the Context

At this point in the operation, restore the context of the SDMA. It can be different from the original context located in memory, and the content previously saved into the debugging application via the OnCE.

The example found hereshows how it is possible to modify the current channel context.

### Modifying the Current Channel Context

```
dmov(Loop_data); // put Loop former value into GReg[1]
exec_once("mov GReg0, GReg1"); // copy to GReg[0]
dmov(PC_data); // put PC former value into GReg[1]
exec_once("mov GReg2, GReg1"); // copy to GReg[2]
dmov(ctx_base_addr); // put channel context base address into
GReg[1]
exec_once("st GReg0, (GReg1,1)"); // restore Loop context
exec_once("st GReg2, (GReg1,0)"); // restore PC context
```

Once the context in memory is the desired context (with or without applying the previous instruction sequence), it can be restored to the *real* PC and loop registers in the SDMA core:

```
exec_once("cpShReg"); // restore flags and PC & loop related registers
```

After this command, the SDMA core PC, RPC, SPC, EPC registers, as well as the flags contain the same data as what is stored in the context RAM for the current channel.

The following example shows how to restore the context of general registers GReg[0], GReg[1], GReg[2] and GReg[3].

### Restoring the General Register Context



```

dmov(GReg3_data); // put GReg[3] restore value in GReg[1]
exec_once("mov GReg3, GReg1"); // restore GReg[3]
dmov(GReg2_data); // put GReg[2] restore value in GReg[1]
exec_once("mov GReg2, GReg1"); // restore GReg[2]
dmov(GReg0_data); // put GReg[0] restore value in GReg[1]
exec_once("mov GReg0, GReg1"); // restore GReg[0]
dmov(GReg1_data); // restore GReg[1]

```

At this point, it is possible to restart the normal program execution.

### NOTE

Every SDMA core general register value can be modified by a mov instruction, which makes modification of these registers easy during debug. Unfortunately, there is no such instruction as a mov to directly modify the contents of either PCU register or flag (PC, RPC, SPC, EPC, T, LM, SF, or DF). The cpShReg instruction is meant to provide a means for changing these register contents via the context memory.

#### 41.4.14.5.4 Accessing the Memory

In the example shown here, it is assumed that the SDMA context is entirely saved. If true, it is permissible to modify the general purpose registers during debugging activity.

To perform a memory read access, the target address is stored via the OnCE in GReg[1], then the load instruction is executed on the SDMA (the data loaded from the memory overwrites the address contained in GReg[1]), and then the result value is read back via the OnCE.

```

macro READ:          dmov(target_addr); // put the target
address in GReg[1]   exec_once("ld GReg1, (GReg1,0)"); // execute the
load instruction     res_data = dmov(-); // exports the result
data value

```

For a memory write access, the target address is written in GReg[0], and the value to store is written in GReg[1]. Then the store instruction is executed on the SDMA.

```

macro WRITE:          dmov(target_addr); // puts the
target address in GReg[1] exec_once("mov GReg0, GReg1"); // puts the target
address in GReg[0]     dmov(target_data); // puts the target
data in GReg[1]        exec_once("st GReg1, (GReg0,0)"); // performs the
store operation

```

This sequence is shown as an example; however, many other sequences are possible.

### NOTE

This sequence of commands can also be applied to memory-mapped registers.

#### 41.4.14.5.5 Resuming Program Execution

Before resuming program execution, it is assumed that the SDMA context is properly restored. There are two ways to restart the SDMA.

Start by executing the last instruction fetched before entering debug mode, as follows.

```
run_core(-); // resume execution from where we stopped before
```

If necessary, restart the execution from a different address. In this case, use the `exec_core` command. The data field provided with this command must be the encoding of a jump instruction.

```
exec_core("jmp start_addr"); // rerun the SDMA from another address
```

In these two examples, the SDMA exits debug mode and keeps executing the code fetched from the memory.

#### 41.4.14.5.6 Single Stepping in RAM

To execute a program step-by-step from the RAM, insert software breakpoints in the program flow at appropriate places so that the SDMA only executes one instruction before returning to debug mode.

First, read the next instruction to execute in the RAM. Then, depending on the value of this instruction, compute the address where a software breakpoint instruction should be inserted. The instruction at the corresponding address must be saved, and, the software breakpoint instruction is inserted. After restarting the SDMA, there is only one instruction executed before meeting the software breakpoint.

The following example shows the macro functions `READ` and `WRITE`, which correspond to the sequence of commands (described above) used to access the memory.

#### NOTE

The data read from the memory are 32-bit values, while the instructions are 16-bit values only. This is why it is best to only use addresses divided by two when accessing the memory.

#### READ and WRITE Macro Functions

```
next_instr = READ(run_addr/2); // read the next instruction to execute
// the tool now has to compute the address where the breakpoint
// instruction should be inserted, this address is the "bkpt_addr"
instr_save = READ(bkpt_addr/2); // save the instruction before
overwriting // store the bkpt instruction
STORE("bkpt instruction",bkpt_addr/2);
in memory
exec_core("jmp run_addr"); // rerun the SDMA
rstatus(-); // wait for the SDMA to enter debug mode
...
rstatus(-);
```

```
STORE(instr_save,bpkt_addr/2);  
overwritten
```

```
// restore the instruction
```

In case of branched conditional instructions, a breakpoint instruction should be written at the two possible target addresses.

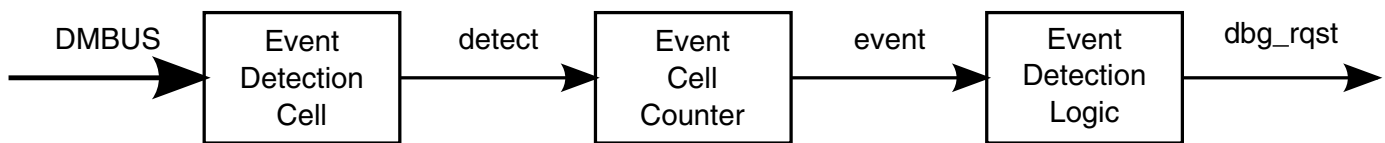
#### 41.4.14.5.7 Single Stepping in ROM

No single-step mechanism is supported in ROM. The program code can be loaded in the RAM, where the single-step mechanism can be executed.

#### 41.4.14.6 OnCE Event Detection Unit

The event detection unit watches signals from the data memory bus (DMBUS), which the SDMA core uses to access its RAM, ROM, and memory mapped registers.

A debug request is sent to the OnCE controller when user-defined conditions on address and/or data values are true.

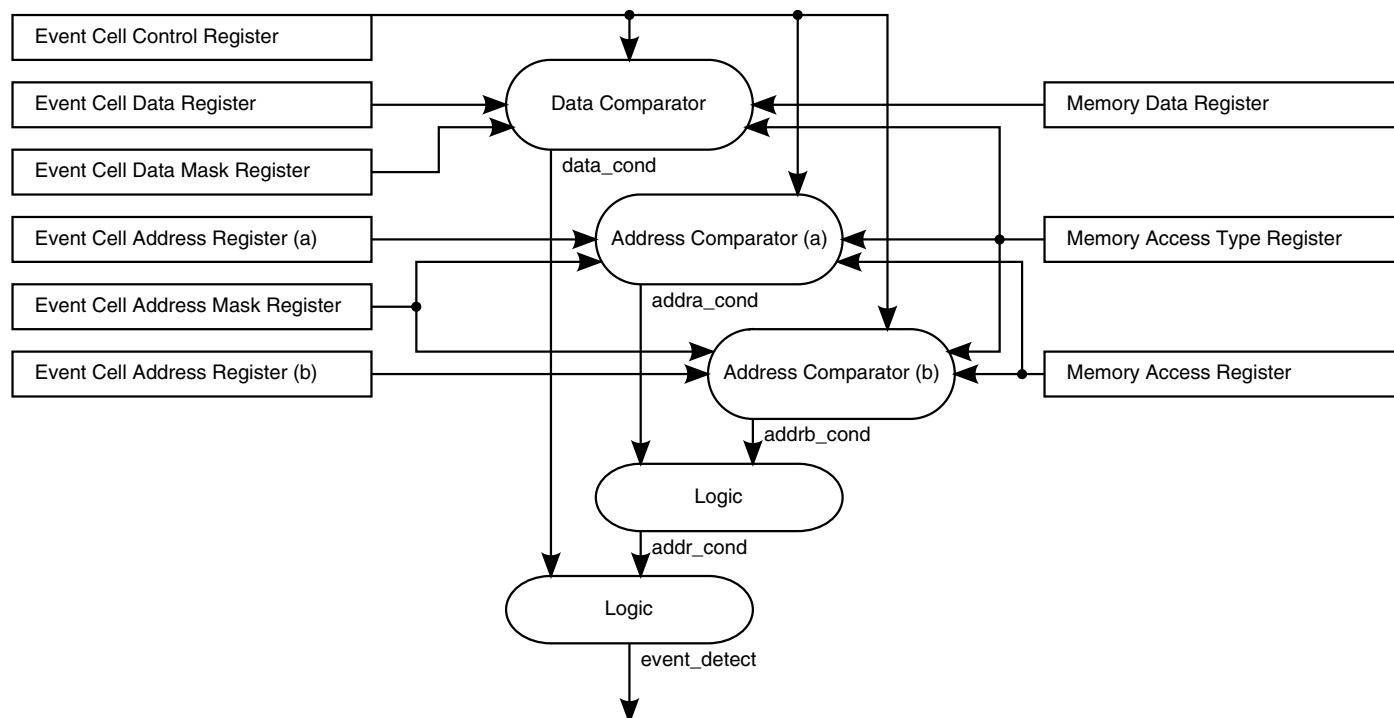


**Figure 41-13. Event Detection Unit**

A counter, provided with the detection cell, is decreased after an event detection. A debug request is sent to the core only when the counter reaches the value of 0. It is possible to disable the use of the counter if a debug request has to be generated after each event detection.

The event cell is the basic block that supports hardware breakpoints on an address value and/or data values coming from the SDMA memory bus. The trigger condition that generates the debug request is a mixed condition based on those values.

The following figure shows the event cell architecture. The event cell contains the address (stored in the memory address register) and the data (stored in the memory data register) used during the last memory access. There are some user-defined reference values located in memory mapped registers—the event cell addresses, the event cell address mask, the event cell data, and the event cell data mask. These registers are accessed by standard load/store instructions just like regular memory locations.



**Figure 41-14. Event Cell Architecture**

To define a memory breakpoint, three conditions are taken into account: The first two conditions are comparisons of the current memory address with user-defined reference addresses (these conditions are called addressA and addressB). The third condition consists of a comparison between the data received on the DMBUS and a user-defined reference data (this condition is called data). An intermediate address condition is set to express a dependency between addressA and addressB conditions.

#### 41.4.14.7 Clock Gating and Reset

This section details how to use the clocks and handle the reset signals.

##### 41.4.14.7.1 Clocks

Because the SDMA uses clock gating to save power, it is necessary to disable the clock gating and force the clocks to be enabled when using the OnCE.

When the OnCE is accessed through its JTAG interface, clock gating must be disabled outside the SDMA via a dedicated SDMA input port `clk_gating_off`. The reason why detection is not performed automatically by the SDMA internal hardware is that it would cost power to monitor activity on the JTAG interface.

When the OnCE is accessed through the ARM platform Control interface, clock gating is automatically turned off. This is done when bit 0 of the ONCE\_ENB register (see [OnCE Enable \(SDMAARM\\_ONCE\\_ENB\)](#)) is set. A write access to this register is possible even when the OnCE clock is not running. If the ARM platform access is used, the bit in the ONCE\_ENB register must be set before any attempt to access any other OnCE register.

#### 41.4.14.7.2 Resets

The OnCE reset is different from the SDMA main reset.

Normally, activating the SDMA reset while keeping the OnCE reset inactive (when possible) enables you to reset the core without having to reprogram the OnCE.

### 41.4.14.8 Real Time Features

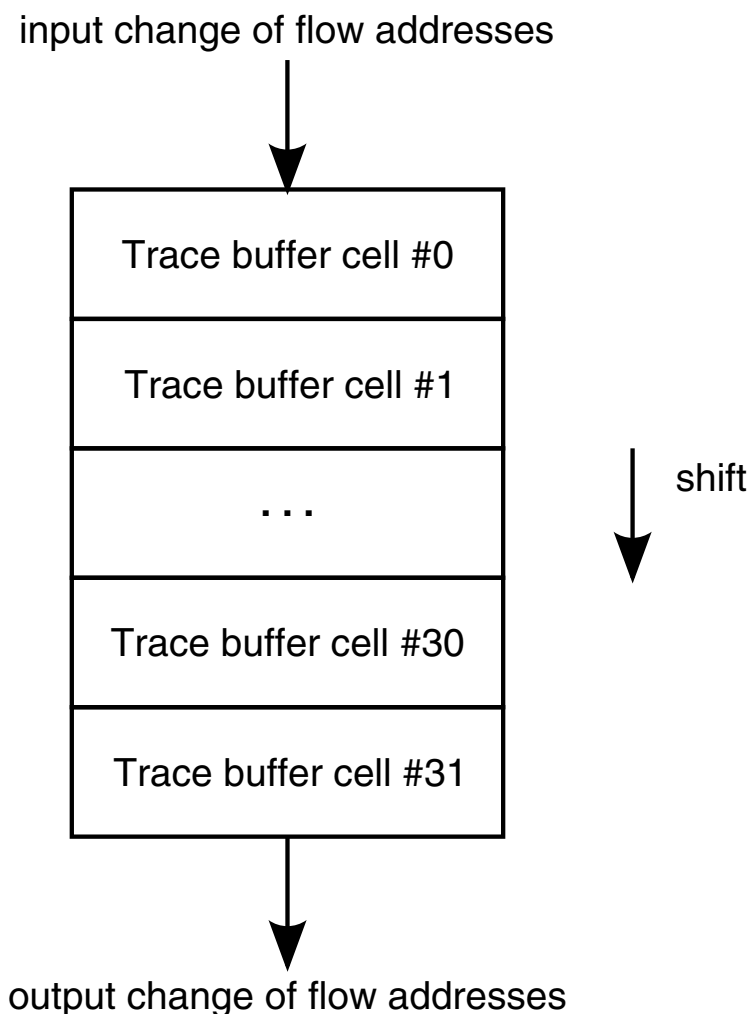
To rebuild the skeleton of a program execution, it is necessary to store the addresses of the program instructions where jumps are taken: A trace buffer is therefore provided. A real time buffer has also been added to receive data values written during a program execution.

The content of this register may be exported through JTAG ports without stopping the core.

#### 41.4.14.8.1 Trace Buffer

The Trace Buffer is a 32-stage buffer that contains appropriate information to identify the 32 last changes of flow detected during a program execution.

The following figure shows an overview of the Trace Buffer.



**Figure 41-15. Trace Buffer**

Each cell of the trace buffer contains two reference addresses and a flag. The flag is set when the addresses stored in the cell correspond to a valid change of flow; otherwise, the flag is cleared. The three most significant bits are unused.

After every change of flow detection, the address of current instruction and the address of the target instruction are stored at the top of the Trace Buffer (cell #0). The flag in the cell is set to indicate that a valid change of flow was detected. Former cell values are shifted one level down. The Trace Buffer contains the 32 last changes of flow. All the flags are reset on a software or a hardware reset, and after each transition from debug mode to user mode.

A memory mapped register of SDMA core, the Trace Buffer register (TB), is provided to read the content of the Trace Buffer. This operation should be done in debug mode. Performing a read access to the Trace Buffer register returns the content of the bottom of the Trace Buffer (cell #31). After every read access, the trace buffer is shifted one level down, and the flag at the top of the trace buffer is cleared.

A typical OnCE command sequence that retrieves the oldest change-of-flow information is as follows:

```
exec_once("mov r1, TB");           // stores the oldest change-of-flow in
GReg1                               // retrieves GReg1 contents
dmov(-);
```

This sequence requires the SDMA to be put in debug mode.

#### 41.4.14.8.2 Real Time Buffer

The Real Time Buffer register (RTB) is a memory mapped register that can be accessed as a regular memory location by the SDMA core during program execution. This register is located in the OnCE.

Executing an `rbuffer` command (see [The OnCE Controller](#) for further details) exports the content of this register through JTAG ports.

When a write access is performed at the memory location corresponding to the RTB, the receive flag (for example, the RCV bit) is set in the OnCE Status Register (OSR). This flag is cleared at the end of the execution of a `rbuffer` command.

#### NOTE

Every write access to the RTB memory location updates the RTB register even if the RCV flag is set. The RTB is cleared on a JTAG reset.

#### 41.4.14.8.3 Emulation Pin

The `debug_matched_event` emulation pin reflects the matching condition status detected by the Event Detection Unit.

Since it can be necessary to detect conditions without triggering debug requests, it is possible to disable the generation of debug requests by the Event Detection Unit and still have the matching condition available on the emulation pin. This can be done by clearing the EN flag in the ECTL register.

#### 41.4.14.8.4 Real-Time Debug Outputs

The table found here shows the debug signals that are available at the SDMA boundaries. Their availability at chip boundaries depends on the project.

**Table 41-41. Real-Time Debug Output Pins**

Pin	Description
debug_core_state[3:0]	<p>The core_state bits reflect the state of the SDMA core.</p> <ul style="list-style-type: none"> <li>• The "Program" state is the usual instruction execution cycle.</li> <li>• The "Data" state is inserted when there are wait-states during a load or a store on the data bus (ld or st).</li> <li>• The "Change of Flow" state is the second cycle of any instruction that breaks the sequence of instructions (jumps and channel switching instructions).</li> <li>• The "Change of Flow in Loop" state is used when an error causes a hardware loop exit.</li> <li>• The "Debug" state means the SDMA is in debug mode.</li> <li>• The "Functional Unit" state is inserted when there are wait-states during a load or a store on the functional units bus (ldf or stf).</li> <li>• In "Sleep" modes, no script is running (this is the core idle state); the "after Reset" is slightly different because no context restoring phase will happen when a channel is triggered: The script located at address 0 is executed (boot operation).</li> <li>• The "in Sleep" states are the same as above except they do not have any corresponding channel: they are used when entering debug mode after reset; the reason is that it is necessary to return to the "Sleep after Reset" state when leaving debug mode.</li> </ul> <p>0 Program  1 Data  2 Change of Flow  3 Change of Flow in Loop  4 Debug  5 Functional Unit  6 Sleep  7 Context Switch Saving Channel  8 Program in Sleep  9 Data in Sleep  10 Change of Flow in Sleep  11 Change of Flow in Loop in Sleep  12 Debug in Sleep  13 Functional Unit in Sleep  14 Sleep after Reset  15 Context Switch Restoring Channel</p>
debug_yield	<p>Pulse that is active when a yield (done 0) or a yieldge (done 1) instruction is executed.</p> <p>0 -  1 yield/yieldge executed</p>

*Table continues on the next page...*



**Table 41-41. Real-Time Debug Output Pins (continued)**

Pin	Description
debug_core_run	Active when the SDMA core is executing instructions. 0 Debug or sleep mode 1 Run mode
debug_event_channel_sel	Indicates if debug_event_channel displays current channel or last received event 0- debug_event_channel[5:0] gives the number of the current channel 1- debug_event_channel[5:0] gives the number of the last received event
debug_event_channel[5:0]	Gives the number of any DMA request as soon as it is received or the number of the current channel.  The value of debug_event_channel_sel indicates if debug_event_channel displays the current channel or last received event. The signal debug_event_channel_sel must be observed to determine what information is provided on debug_event_chanel at any given time.
debug_pc[13:0]	Program Counter value; it has a meaning when the core is in run mode.
debug_mode	Set when the core is in debug. 0 - 1 Core is in debug
debug_bus_error	Set when an error was received during a load or a store (ld, st, ldf, or stf instruction) and registered in SF or DF flag. 0 No error during last load/store 1 Error during last load/store

*Table continues on the next page...*

**Table 41-41. Real-Time Debug Output Pins (continued)**

Pin	Description
debug_bus_device[4:0]	Indicates the device or functional unit that is accessed by the current instruction. The debug_bus_device output is always valid when in sleep mode, debug mode, or executing any instruction that does not access the functional units or the memory mapped devices, "no access" is output.
	0 No access
	1 MSA
	2 MDA
	3 MD
	4 MS
	5 PSA
	6 PDA
	7 PD
	8 PS
	9 RESERVED
	10 RESERVED
	11 RESERVED
	12 RESERVED
	13 CA
	14 CS
	15 Reserved
	16 Memory (RAM or ROM)
	17 Memory mapped register
	18 Peripheral #1
	19 Peripheral #2
	20 Peripheral #3
	21 Peripheral #4
	22 Peripheral #5
	23 Peripheral #6
	24 Peripheral #7
	25 Peripheral #8
	26 Peripheral #9
	27 Peripheral #10
	28 Peripheral #11
	29 Peripheral #12
	30 Peripheral #13
	31 Peripheral #14

*Table continues on the next page...*

**Table 41-41. Real-Time Debug Output Pins (continued)**

Pin	Description
debug_bus_rwb	Indicates the direction of the access given by debug_bus_device 0 Write access (st or stf) 1 Read access (ld or ldf)
debug_matched_dmbus	Pulse indicating the OnCE event detection unit has detected a match on the data bus during an access to memory (RAM or ROM), a memory mapped register or a peripheral that is hooked to the SDMA. 0 - 1 data bus match detected
debug_rtbuffer_write	Pulse indicating when the real-time buffer is written by the core. 0 - 1 RTB was modified
debug_evt_chn_lines[7:0]	Eight lines that generate short pulses when DMA requests are received or channels are (re)started. Every line is controlled through two parameters defined in registers <a href="#">Cross-Trigger Events Configuration Register 1 (SDMAARM_XTRIG_CONF1)</a> (as described in <a href="#">SDMAARM</a> ). The following two parameters are available for every line: <ul style="list-style-type: none"> <li>• CNF-Indicates what is monitored on the line: 0 for a channel start, 1 for a DMA request reception</li> <li>• NUM[ 5:0]-Gives the number of the DMA request or channel to monitor</li> </ul>

The matched\_event emulation pin reflects the matching condition status detected by the Event Detection Unit. Because it can be necessary to detect conditions without triggering debug requests, it is possible to disable the generation of debug requests by the Event Detection Unit and still have the matching condition available on the emulation pin. This can be done by clearing the EN flag in the ECTL register.

All real-time debug outputs are disabled by default (for example, they are stuck to 0) to avoid power consumption when they are not used. They are enabled when bit 11 (RTDOBS) of the [Configuration Register \(SDMAARM\\_CONFIG\)](#) is set. Signals provided to the system JTAG controller for SDMA debug mode status will also be enabled when the *clk\_gating\_off* input is asserted.

## 41.5 Instruction Set

### 41.5.1 Instruction Encoding

This section presents a short summary of the instruction codes. All context switch instructions are listed for information only; they cannot function properly out of the context switch routine.

x...x - don't care  
 rrr - destination/source general register  
 sss - additional source general register  
 bbb - general register used as address base register  
 dddd - address displacement  
 nnnnn - bit number  
 uuuuuuuu - function unit command bits  
 pppppppp - branch displacement (signed)  
 iiiiii - 8-bit immediate  
 jjj - control bit to clear  
 ff - flag to clear  
 00000jjj00000000 - done (done,yield,wait)  
 00000jjj00000001 - notify  
 00000xxx00000010 - reserved  
 00000xxx00000011 - reserved  
 00000xxx00000100 - reserved  
 0000000000000101 - softBkpt  
 0000000100000101 - reserved  
 0000001000000101 - reserved  
 0000001100000101 - reserved  
 0000010000000101 - reserved  
 0000010100000101 - reserved  
 0000011000000101 - reserved  
 0000011100000101 - reserved  
 0000000000000110 - ret  
 0000000100000110 - reserved  
 0000001000000110 - reserved  
 0000001100000110 - reserved  
 0000010000000110 - reserved  
 0000010100000110 - reserved  
 0000011000000110 - reserved  
 0000011100000110 - reserved  
 000000ff00000111 - clrf ff  
 0000010000000111 - reserved  
 0000010100000111 - reserved  
 0000011000000111 - reserved  
 0000011100000111 - illegal  
 00000rrr00001000 - jmp r  
 00000rrr00001001 - jsrr  
 00000rrr00001010 - ldrpc r  
 00000rrr00001011 - reserved  
 00000rrr000011xx - reserved  
 00000rrr00010000 - revb  
 00000rrr00010001 - revblo  
 00000rrr00010010 - rorb  
 00000rrr00010011 - reserved  
 00000rrr00010100 - rorl  
 00000rrr00010101 - lsr1  
 00000rrr00010110 - asr1  
 00000rrr00010111 - lsl1  
 00000rrr001nnnnn - bclri r,n  
 00000rrr010nnnnn - bseti r,n  
 00000rrr011nnnnn - btsti r,n  
 00000xxx10000xxx - reserved  
 00000rrr10001sss - mov  
 00000rrr10010sss - xor  
 00000rrr10011sss - add  
 00000rrr10100sss - sub  
 00000rrr10101sss - or  
 00000rrr10110sss - andn

```

00000rrrr10111sss      - and
00000rrrr11000sss      - tst
00000rrrr11001sss      - cmpeq
00000rrrr11010sss      - cmplt
00000rrrr11011sss      - cmphs
0000011011100000      - reserved
0000011011100001      - reserved
0000011011100010      - cpShReg
0000011011100011      - reserved
0000011011100100      - reserved
0000011011100101      - reserved
0000011011100110      - reserved
0000011011100111      - reserved
00000xxx11101xxx      - reserved
00000xxx11110xxx      - reserved
00000xxx11111xxx      - reserved
00001rrriiiiiiii      - ldi r,i
00010rrriiiiiiii      - xori r,i
00011rrriiiiiiii      - addi r,i
00100rrriiiiiiii      - subi r,i
00101rrriiiiiiii      - ori r,i
00110rrriiiiiiii      - andni r,i
00111rrriiiiiiii      - andi r,i
01000rrriiiiiiii      - tsti r,i
01001rrriiiiiiii      - cmpeqi r,i
01010rrrddddd bbb      - ld r,(d,b)
01011rrrddddd bbb      - st r,u
01100rrruuuuuuuu      - ldf r,u
01101rrruuuuuuuu      - stf r,u
011100xxxxxxxxxx      - reserved
011101xxxxxxxxxx      - reserved
011110ffnnnnnnnn      - Loop ff flags are reset
01111100pppppppp      - bf pc=pc+signed(pppppppp)+1
01111101pppppppp      - bt pc=pc+signed(pppppppp)+1
01111110pppppppp      - bsf pc=pc+signed(pppppppp)+1
01111111pppppppp      - bdf pc=pc+signed(pppppppp)+1
10aaaaaaaaaaaaaaaa      - jmp absolute
11aaaaaaaaaaaaaaaa      - jsr absolute

```

## 41.5.2 SDMA Instruction Set

This section describes all the useful instructions from the SDMA set.

**Table 41-42. SDMA Instruction List**

Inst ruct ion	Description	Page
ADD	Addition	<a href="#">ADD (Addition)</a>
ADD I	Add with Immediate Value	<a href="#">ADDI (Add with Immediate Value)</a>
AND	Logical AND	<a href="#">AND (Logical AND)</a>
AND I	Logical AND with Immediate Value	<a href="#">ANDI (Logical AND with Immediate Value)</a>
AND N	Logical AND NOT	<a href="#">ANDN (Logical AND NOT)</a>
AND NI	Logical AND with Negated Immediate Value	<a href="#">ANDNI (Logical AND with Negated Immediate Value)</a>

*Table continues on the next page...*

**Table 41-42. SDMA Instruction List  
(continued)**

Inst ruct ion	Description	Page
ASR 1	Arithmetic Shift Right by 1 Bit	<a href="#">ASR1 (Arithmetic Shift Right by 1 Bit)</a>
BCL RI	Bit Clear Immediate	<a href="#">BCLRI1 (Bit Clear Immediate)</a>
BDF	Conditional Branch if Destination Fault	<a href="#">BDF (Conditional Branch if Destination Fault)</a>
BF	Conditional Branch if False	<a href="#">Functional Units Programming Model</a>
BSE TI	Bit Set Immediate	<a href="#">BSETI (Bit Set Immediate)</a>
BSF	Conditional Branch if Source Fault	<a href="#">BSF (Conditional Branch if Source Fault)</a>
BT	Conditional Branch if True	<a href="#">BT (Conditional Branch if True)</a>
BTS TI	Bit Test immediate	<a href="#">BTSTI (Bit Test immediate)</a>
CLR F	Clear ARM platform flags	<a href="#">CLRf (Clear ARM platform flags)</a>
CMP EQ	Compare for Equal	<a href="#">CMPEQ (Compare for Equal)</a>
CMP EQI	Compare with Immediate for Equal	<a href="#">CMPEQI (Compare with Immediate for Equal)</a>
CMP HS	Compare for Higher or Same	<a href="#">CMPHS (Compare for Higher or Same)</a>
CMP LT	Compare for Less Than	<a href="#">CMPLT (Compare for Less Than)</a>
cpSh Reg	Update Context of PCU Registers and Flags	<a href="#">cpShReg (Update Context of PCU Registers and Flag)</a>
DON E	DONE, Yield	<a href="#">DONE (DONE, Yield)</a>
ILLE GAL	ILLEGAL Instruction	<a href="#">ILLEGAL (ILLEGAL Instruction)</a>
JMP	Unconditional Jump Immediate	<a href="#">JMP (Unconditional Jump Immediate)</a>
JMP R	Unconditional Jump	<a href="#">JMPR (Unconditional Jump)</a>
JSR	Unconditional Jump to Subroutine Immediate	<a href="#">JSR (Unconditional Jump to Subroutine Immediate)</a>
JSR R	Unconditional Jump to Subroutine	<a href="#">JSRR (Unconditional Jump to Subroutine)</a>
LD	Load Register	<a href="#">LD (Load Register)</a>
LDF	Load Register from Functional Unit	<a href="#">LDF (Load Register from Functional Unit)</a>
LDI	Load Register with Immediate Value	<a href="#">LDI (Load Register with Immediate Value)</a>
LDR PC	Load from RPC to Register	<a href="#">LDRPC (Load from RPC to Register)</a>
LOO P	Hardware Loop	<a href="#">LOOP (Hardware Loop)</a>

*Table continues on the next page...*

**Table 41-42. SDMA Instruction List  
(continued)**

Inst ruct ion	Description	Page
LSL 1	Logical Shift Left by 1 Bit	LSL1 (Logical Shift Left by 1 Bit)
LSR 1	Logical Shift Right by 1 Bit	LSR1 (Logical Shift Right by 1 Bit)
MOV	Logical Move	MOV (Logical Move)
NOT IFY	Notify to ARM platform	NOTIFY (Notify to ARM platform)
OR	Logical OR	OR (Logical OR)
ORI	Logical OR with Immediate Value	ORI (Logical OR with Immediate Value)
RET	Return from Subroutine	RET (Return from Subroutine)
REV B	Reverse Byte Order	REVB (Reverse Byte Order)
REV BLO	Reverse Low Order Bytes	Reverse Low Order Bytes(REVBLO)
ROR 1	Rotate Right by 1 Bit	ROR1 (Rotate Right by 1 Bit)
ROR B	Rotate Right by 1 Byte	RORB (Rotate Right by 1 Byte)
SOF TBK PT	Software Breakpoint	SOFTBKPT (Software Breakpoint)
ST	Store Register	ST (Store Register)
STF	Store Register in Functional Unit	STF (Store Register in Functional Unit)
SUB	Subtract	SUB (Subtract)
SUB I	Subtract with Immediate	SUBI (Subtract with Immediate)
TST	Test with Zero	TST (Test with Zero)
TSTI	Test Immediate	TSTI (Test Immediate)
XOR	Logical Exclusive OR	XOR (Logical Exclusive OR)
XOR I	Exclusive OR with Immediate	XORI (Exclusive OR with Immediate)

### 41.5.2.1 ADD (Addition)

#### Operation:

$$\text{GReg}[\text{r}] \leftarrow \text{GReg}[\text{s}] + \text{GReg}[\text{r}]$$

$$\text{T} \leftarrow (\text{GReg}[\text{r}] == 0)$$

#### Assembler:

## Instruction Set

Syntax: `add r,s`

Example: `add 0,3`

ADD GReg[3] and GReg[0] and store the result in GReg[0]

CPU Flags: T

Cycles: 1

Description: Performs the ADDition of the source general register *s* and the destination general register *r*, and stores the result in the destination general register *r*. The T flag is set if the result of the operation is 0. It is cleared if the result is not 0.

Instruction Format:

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	0	1	1	s	s	s

Instruction Fields:

rrr / sss - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

### 41.5.2.2 ADDI (Add with Immediate Value)

**Operation:**

$\text{GReg}[r] \leftarrow \text{GReg}[r] + \text{immediate}$

$T \leftarrow (\text{GReg}[r] == 0)$

**Assembler:**

Syntax: `addi r,immediate`

Example: `add 6,112`

ADD GReg[6] and decimal value 112 and store the result in GReg[6]



CPU Flags: T

Cycles: 1

Description: Adds a 0-extended immediate value to a general register; stores the result in the general register. The flag T is set when the result of the operation is 0; otherwise, it is cleared. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	1	r	r	r	i	i	i	i	i	i	i	i

Instruction Fields:

rrr - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

iiiiiii - immediate value:

00000000 - 0

00000001 - 1

...

11111110 - 254

11111111 - 255

### 41.5.2.3 AND (Logical AND)

**Operation:**

$\text{GReg}[r] \leftarrow \text{GReg}[s] \ \& \ \text{GReg}[r]$

**Assembler:**

## Instruction Set

Syntax: `and r,s`

Example: `and 1,2`

AND GReg[1] and GReg[2] and store the result in GReg[1]

CPU Flags: Unaffected

Cycles: 1

Description: Performs the AND of the source general register *s* and the destination general register *r*, and stores the result in the destination general register *r*.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	1	1	1	s	s	s

Instruction Fields:

rrr / sss - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

### 41.5.2.4 ANDI (Logical AND with Immediate Value)

**Operation:**

$\text{GReg}[r] \leftarrow \text{GReg}[r] \& \text{immediate}$

**Assembler:**

Syntax: `andi r,immediate`

Example: `andi 7,45`

AND GReg[7] and decimal value 45 and store the result in GReg[7]

CPU Flags: unaffected

Cycles: 1

**Description:** Performs an AND between a 0-extended immediate value and a general register; stores the result in the general register. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

**Instruction Format**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	1	1	r	r	r	i	i	i	i	i	i	i	i

**Instruction Fields:**

**rrr - register field:**

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

**iiiiiii - immediate value:**

00000000 - 0

00000001 - 1

...

11111110 - 254

11111111 - 255

### 41.5.2.5 ANDN (Logical AND NOT)

**Operation:**

$\text{GReg}[r] \leftarrow \sim \text{GReg}[s] \ \& \ \text{GReg}[r]$

**Assembler:**

Syntax: `andn r,s`

Example: `andn 3,4`

## Instruction Set

AND GReg[3] and NOT GReg[4] (bit inverted) and store the result in GReg[3]

CPU Flags: Unaffected

Cycles: 1

Description: Performs the AND of the negation of the source general register *s* and the destination general register *r*, and stores the result in the destination general register *r*.

Instruction Format:

**Table 41-43. Instruction Format**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	<i>r</i>	<i>r</i>	<i>r</i>	1	0	1	1	0	<i>s</i>	<i>s</i>	<i>s</i>

Instruction Fields:

rrr /sss - destination register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

### 41.5.2.6 ANDNI (Logical AND with Negated Immediate Value)

**Operation:**

$\text{GReg}[r] \leftarrow \text{GReg}[r] \& \sim\text{immediate}$

**Assembler:**

Syntax: `andni r,immediate`

Example: `andni 0,2`

AND GReg[0] and decimal value -3 (inverted 32-bit value 2) and store the result in GReg[0]

CPU Flags: unaffected

Cycles: 1

**Description:** Performs an AND between the negation of a 0-extended 8-bit immediate value and a general register; stores the result in the general register. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

**Instruction Format**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	1	0	r	r	r	i	i	i	i	i	i	i	i

**Instruction Fields:**

**rrr - register field:**

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

**iiiiiii - immediate value:**

00000000 - 0

00000001 - 1

...

11111110 - 254

11111111 - 255

### 41.5.2.7 ASR1 (Arithmetic Shift Right by 1 Bit)

**Operation:**

$\text{GReg}[r] : \{b_{31}, b_{30}, \dots, b_1, b_0\} \leftarrow \text{GReg}[r] : \{b_{31}, b_{31}, b_{30}, \dots, b_1\}$

**Assembler:**

Syntax: `asr1 r`

Example: `asr1 3`

divide by 2 the signed value of GReg[3] and store the result in GReg[3]

## Instruction Set

CPU Flags: Unaffected

Cycles: 1

Description: Shift the bits of any general register to the right and keep the same sign: The left bit (bit 31) is kept untouched.

Instruction Format:

**Table 41-44. Instruction Format**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	1	1	0

Instruction Fields:

rrr - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

### 41.5.2.8 BCLRI1 (Bit Clear Immediate)

**Operation:**

$\text{GReg}[r] : \{b_{31}, \dots, b_{(i+1)}, 0, b_{(i-1)}, \dots, b_0\} \leftarrow \text{GReg}[r] : \{b_{31}, \dots, b_{(i+1)}, b_{(i)}, b_{(i-1)}, \dots, b_0\}$

**Assembler:**

Syntax: `bclri r,i`

Example: `bclri 1,12`

clear bit 12 in GReg[1]

CPU Flags: Unaffected

Cycles: 1

Description: Clear the bit of register r specified by the 5-bit immediate field

## Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	1	i	i	i	i	i

rrr - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

iiii - immediate value:

00000 - 0

00001 - 1

...

11110 - 30

11111 - 31

### 41.5.2.9 BDF (Conditional Branch if Destination Fault)

#### Operation:

if (DF == 1) PC ← PC + 1 + displacement else PC ← PC + 1

#### Assembler:

Syntax: bdf label

Example: bdf LLL

Jump to LLL if DF is set, or go to the next instruction if DF is cleared; the displacement value is calculated by the assembler.

CPU Flags: Unaffected

Cycles: 2 when the branch is done, 1 otherwise

## Instruction Set

**Description:** If flag DF is set, jump to the new address that is calculated by adding the sign-extended 8-bit displacement to the next PC address. If flag DF is cleared, no jump is performed: The next instruction is located at the next PC address.

### Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	1	1	1	p	p	p	p	p	p	p	p

### Instruction Fields:

pppppppp - signed displacement field:

00000000 - 0

00000001 - 1

...

01111110 - 126

01111111 - 127

10000000 - (-128)

10000001 - (-127)

...

11111110 - (-2)

11111111 - (-1)

## 41.5.2.10 BF (Conditional Branch if False)

### Operation:

```
if (T == 0)
```

```
PC ← PC + 1 + displacement
```

```
else
```

```
PC ← PC + 1
```

### Assembler:

Syntax: bf label

Example: bf LLL

Jump to LLL if T is cleared, or go to the next instruction if T is set. The displacement value is calculated by the assembler.

CPU Flags: Unaffected



Cycles: 2 when the branch is done, 1 otherwise

Description: Conditional branch: If flag T is cleared, jump to the new address that is calculated by adding the sign-extended 8-bit displacement to the next PC address. If flag T is set, no jump is performed: The next instruction is located at the next PC address.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	1	0	0	p	p	p	p	p	p	p	p

Instruction Fields:

pppppppp - signed displacement field:

```

00000000 - 0
00000001 - 1
...
01111110 - 126
01111111 - 127
10000000 - (-128)
10000001 - (-127)
...
11111110 - (-2)
11111111 - (-1)

```

### 41.5.2.11 BSETI (Bit Set Immediate)

**Operation:**

$$\text{GReg}[r] : \{b_{31}, \dots, b_{(i+1)}, 1, b_{(i-1)}, \dots, b_0\} \leftarrow \text{GReg}[r] : \{b_{31}, \dots, b_{(i+1)}, b_{(i)}, b_{(i-1)}, \dots, b_0\}$$

**Assembler:**

Syntax: `bseti r,i`

Example: `bseti 6,5`

Set bit 5 in GReg[6]

CPU Flags: Unaffected

Cycles: 1

Description: Sets bit number i in the selected General Register.

## Instruction Set

### Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	1	0	i	i	i	i	i

### Instruction Fields:

#### rrr - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

#### iiii - bit number field:

00000 - 0

00001 - 1

...

11110 - 30

11111 - 31

## 41.5.2.12 BSF (Conditional Branch if Source Fault)

### Operation:

if (SF == 1) PC ← PC + 1 + displacement else PC ← PC + 1

### Assembler:

Syntax: bsf label

Example: bsf LLL

Jump to LLL if SF is set, or go to the next instruction if SF is cleared. The displacement value is calculated by the assembler.

CPU Flags: Unaffected

Cycles: 2 when the branch is done, 1 otherwise

**Description:** Conditional branch: If flag SF is set, jump to the new address that is calculated by adding the sign-extended 8-bit displacement to the next PC address. If flag SF is cleared, no jump is performed: The next instruction is located at the next PC address.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	1	1	0	p	p	p	p	p	p	p	p

Instruction Fields:

pppppppp - signed displacement field:

```
00000000 - 0
00000001 - 1
...
01111110 - 126
01111111 - 127
10000000 - (-128)
10000001 - (-127)
...
11111110 - (-2)
11111111 - (-1)
```

### 41.5.2.13 BT (Conditional Branch if True)

#### Operation

```
if (T == 1)
    PC ← PC + 1 + displacement
else
    PC ← PC + 1
```

#### Assembler

```
Syntax: bt label

bt LLL
```

Jump to LLL if T is set, or go to the next instruction if T is cleared. The displacement value is calculated by the assembler.

## Instruction Set

CPU Flags: Unaffected

Cycles: 2 when the branch is done, 1 otherwise

Description: Conditional branch: If flag T is set, jump to the new address that is calculated by adding the sign-extended 8-bit displacement to the next PC address. If flag T is cleared, no jump is performed: The next instruction is located at the next PC address.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	1	0	1	p	p	p	p	p	p	p	p

pppppppp - signed displacement field:

```
00000000 - 0
00000001 - 1
...
01111110 - 126
01111111 - 127
10000000 - (-128)
10000001 - (-127)
...
11111110 - (-2)
11111111 - (-1)
```

### 41.5.2.14 BTSTI (Bit Test immediate)

#### Operation:

$T \leftarrow \text{GReg}[r] : b(i)$

#### Assembler:

Syntax: `btsti r,i`

Example: `btsti 2,29`

Test bit 29 in GReg[2] and copy its value in flag T

CPU flags: T

Cycles: 1

Description: T is loaded with the value of bit number i from the selected general register.

## Instruction Format:

**Table 41-45. Instruction Format**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	1	1	i	i	i	i	i

## Instruction Fields:

## rrr - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

## iiii - bit number field:

0000 - 0

0001 - 1

...

11110 - 30

11111 - 31

**41.5.2.15 CLRF (Clear ARM platform flags)****Operation:**

if (ff%2 == 0)

SF ← 0

if (ff/2 == 0)

DF ← 0

**Assembler:**

Syntax: clrf ff

Example: clrf 2

## Instruction Set

Clear flag SF and keep flag DF unchanged

CPU Flags: SF, DF

Cycles: 1

Description: Clears a selection of the ARM platform fault flags: SF, DF, both SF and DF or none can be cleared.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	f	f	0	0	0	0	0	1	1	1

Instruction Fields:

ff - flags field:

00 - clear SF and clear DF

01 - clear DF

10 - clear

SF 11 - no clear

## 41.5.2.16 CMPEQ (Compare for Equal)

**Operation:**

$T \leftarrow (GReg[s] == GReg[r])$

**Assembler:**

Syntax: cmpeq r,s

Example: cmpeq 7,5

Compare GReg[7] and GReg[5] and set flag T if they are equal

CPU flags: T

Cycles: 1

Description: Subtracts the destination general register *r* from the source general register *s*, and sets T if the result is 0, clears T if the result is not 0.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	1	0	0	1	s	s	s

**Instruction Fields:****rrr / sss - register field:**

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

**41.5.2.17 CMPEQI (Compare with Immediate for Equal)****Operation:** $T \leftarrow (GReg[r] == \text{immediate})$ **Assembler:**

Syntax: cmpeqi r,immediate

Example: cmpeqi 2,13

Compare GReg[2] and decimal value 13 and set flag T if they are equal

**CPU Flags:** T**Cycles:** 1

**Description:** Subtracts the 0-extended 8-bit immediate value from the general register, and sets T if the result is 0, clears T if the result is not 0. The immediate value is the low-order byte of the instruction.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	1	r	r	r	i	i	i	i	i	i	i	i

**Instruction Fields:****rrr - destination register field:**

000 - GReg[0]

## Instruction Set

001 - GReg[1]  
010 - GReg[2]  
011 - GReg[3]  
100 - GReg[4]  
101 - GReg[5]  
110 - GReg[6]  
111 - GReg[7]

iiiiiii - immediate value:

00000000 - 0  
00000001 - 1  
...  
11111110 - 254  
11111111 - 255

### 41.5.2.18 CMPHS (Compare for Higher or Same)

#### Operation:

$T \leftarrow (GReg[r] \geq GReg[s])$

#### Assembler:

Syntax: `cmphs r,s`

Example: `cmphs 0,1`

Compare GReg[0] and GReg[1] and set flag T if GReg[0] is higher than or equal to GReg[1]

CPU Flags: T

Cycles: 1

Description: Compares the destination general register *r* and the source general register *s*, and sets T if the destination general register *r* is higher than or equal to the source general register *s*, clears T otherwise. The comparison is unsigned.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	1	0	1	1	s	s	s



**Instruction Fields:****rrr / sss - register field:**

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

**41.5.2.19 CMPLT (Compare for Less Than)****Operation:** $T \leftarrow (GReg[r] < GReg[s])$ **Assembler:**Syntax: `cmplt r,s`Example: `cmplt 7,4`

Compare GReg[7] and GReg[4] and set flag T if GReg[7] is lower than GReg[4]

CPU Flags: T

Cycles: 1

Description: Compares the destination general register *r* and the source general register *s*, and sets T if the destination general register *r* is lower than the source general register *s*, clears T otherwise. The comparison is signed.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	1	0	1	0	s	s	s

**rrr / sss - register field:**

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

## Instruction Set

011 - GReg[3]  
100 - GReg[4]  
101 - GReg[5]  
110 - GReg[6]  
111 - GReg[7]

### 41.5.2.20 cpShReg (Update Context of PCU Registers and Flag)

#### Assembler:

Syntax: cpShReg

CPU Flags: none

Cycles: 1

Description: SF, RPC, T, PC, LM, EPC, DF, and SPC registers are updated according to the value of their corresponding bits in the context memory. This instruction must only be used in debug mode via the OnCE. It reverses the done 5 operation.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	1	0	1	1	1	0	0	0	1	0

### 41.5.2.21 DONE (DONE, Yield)

#### Operation:

```
if (jjj&6 == 2) HE[CCR] ← 0
```

```
if (jjj == 3) HI[CCR] ← 1
```

```
if (jjj == 4) EP[CCR] ← 0
```

```
if ((jjj == 0) && (NCP > CCP)) CCR ← NCR
```

```
else if ((jjj == 1) && (NCP >= CCP))
```

```
CCR ← NCR
```

```
else
```

```
CCR ← NCR
```

(CCR stands for Current Channel Register; NCR stands for Next Channel Register)

**Assembler:**

Syntax: done jjj

Example: done 3

Clear HE bit for the current channel, send an interrupt to the ARM platform for the current channel and reschedule.

CPU Flags: Unaffected

Cycles: Variable if a context switch is done, 1 otherwise

Description: Clears one of the channel enabling bits (HE or EP for the corresponding channel number) if required. Sends an interrupt to the corresponding ARM platform by setting the appropriate flag, if required (HI for the corresponding channel number). Reschedules according to the mode and the NCP (Next Channel Priority) and CCP (Current Channel Priority) values. According to the scheduling decision, the NCR (Next Channel Register) is copied to the CCR (Current Channel Register) and channel contexts are switched. If several channels with the same highest priority are pending, they are ordered by their number from 31 down to 0. The higher number is selected (for example, channel 26 is selected if channels 3, 12, 14, and 26 with the same highest priority are pending). If no flag is modified, the reschedule can allow the replacement of the current channel by another channel with a priority strictly greater than the current channel priority (yield). Or, it can allow the replacement of the current channel by another channel with a priority greater than or equal to the current channel priority (yieldge). In the latter case, the selected channel will always be the first one with the same priority, starting from channel number 31 down to channel 0 (the current channel does not belong to the set of selectable channels).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	j	j	j	0	0	0	0	0	0	0	0

jjj - Channel Flags field:

000 - No channel flags affected: Reschedule only if the next channel priority is greater than current channel priority (yield)

001 - No channel flags affected: Reschedule only if the next channel priority is greater than or equal to the current channel priority (yieldge)

010 - Clear HE for the current channel and reschedule 011 - Clear HE, set HI for the current channel and reschedule 100 - Clear EP for the current channel and reschedule

101 - Reserved for debug to copy relevant registers into context memory

110 - RESERVED

111 - RESERVED

For the scheduling rules, refer to [Scheduler Functional Description](#). Every possible done instruction is further described as follows:

- done 0/yield is executed by a channel script when it accepts preemption by a higher priority channel;
- done 1/yieldge is executed by a channel script when it accepts preemption by a higher priority channel and it also accepts a roll-up with other channels that have the same priority;
- done 2 is executed by a channel script that was triggered by a ARM platform start via the [Channel Start \(SDMAARM\\_HSTART\)](#) register, when its task is completed and it requires termination;
- done 3 is executed by a channel script that was triggered by a ARM platform start via the [Channel Start \(SDMAARM\\_HSTART\)](#) register, when its task is completed, it requires termination and it needs to trigger an interrupt to the ARM platform upon closure;
- done 4 is executed by a channel script that was triggered by a DMA request, when its task is completed and it requires termination;
- done 5 is used in debug mode only; it copies the PCU registers and flags to the context memory of the current channel;

#### 41.5.2.22 ILLEGAL (ILLEGAL Instruction)

##### Operation:

$PC \leftarrow 0001$

##### Assembler:

Syntax: `illegal`

CPU Flags: Unaffected

Cycles: 2

Description: Jumps to the Illegal instruction routine located at address 0001. All unauthorized instructions result in an Illegal instruction behavior; however, the ILLEGAL instruction must be used to guarantee software compatibility with future versions of the SDMA.

Instruction Format

**Table 41-46. Instruction Format**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	1	1	1	0	0	0	0	0	1	1	1

### 41.5.2.23 JMP (Unconditional Jump Immediate)

#### Operation:

$PC \leftarrow \text{absolute\_address}$

#### Assembler:

Syntax: `jmp label`

Example: `jmp LLL`

The assembler translates the label to the exact address

CPU Flags: Unaffected

Cycles: 2

Description: Jumps to the absolute address contained the lower 14 bits of the instruction (the PC is a 14-bit register).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	0	a	a	a	a	a	a	a	a	a	a	a	a	a	a

aaaaaaaaaaaaaaaa - address field:

0000000000000000 - 0

0000000000000001 - 1

...

1111111111111110 - 16382

1111111111111111 - 16383

### 41.5.2.24 JMPR (Unconditional Jump)

#### Operation:

$PC \leftarrow GReg[r]$

## Instruction Set

### Assembler:

Syntax: `jmp r`

Example: `jmp 0`

Jump to address stored in GReg[0]

CPU Flags: Unaffected

Cycles: 2

Description: Jumps to the absolute address contained in a General Register.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	0	1	0	0	0

### Instruction Fields:

rrr - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

## 41.5.2.25 JSR (Unconditional Jump to Subroutine Immediate)

### Operation:

$RPC \leftarrow PC + 1$

$PC \leftarrow \text{absolute\_address}$

### Assembler:

Syntax: `jsr r`

Example: `jsr LLL`

Jumps to subroutine starting at LLL; the assembler translates the label to exact address

CPU Flags: Unaffected

Cycles: 2

Description: Jumps to the subroutine located at the absolute address contained the lower 14 bits of the instruction (the PC is a 14-bit register).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
1	1	a	a	a	a	a	a	a	a	a	a	a	a	a	a

aaaaaaaaaaaaaa - address field:

0000000000000000 - 0

0000000000000001 - 1

...

111111111111110 - 16382

111111111111111 - 16383

## 41.5.2.26 JSRR (Unconditional Jump to Subroutine)

**Operation:**

$RPC \leftarrow PC + 1$

$PC \leftarrow GReg[r]$

**Assembler:**

Syntax: jsrr r

Example: jsrr 5

Jumps to subroutine located at address stored in GReg[5]

CPU Flags: Unaffected

Cycles: 2

Description: Jumps to the subroutine at address contained in a General Register

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	0	1	0	0	1

**Instruction Fields:****rrr - register field:**

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

### **41.5.2.27 LD (Load Register)**

**Operation:**

$\text{GReg}[r] \leftarrow [\text{GReg}[b] + \text{displacement}]$

if (transfer\_error)

SF  $\leftarrow$  1

else

SF  $\leftarrow$  0

**Assembler:**

Syntax: `ld r, (b, displacement)`

Example: `ld 1, (2, 23)`

Loads data into GReg[1]; the data is located at address obtained by adding decimal value 23 to GReg[2]

CPU Flags: SF

Cycles: 2+n where n is 0 for ROM, RAM or memory mapped registers, and n is the number of wait-states of the peripheral for a peripheral access

Description: Adds a 5-bit 0-extended displacement to a base address in General Register b; the result is the address of the data to fetch on the DM bus. The data received from the bus is stored in the destination General Register r. If an error occurs during the transfer, the flag SF is set, else it is cleared.

Instruction Format



15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	0	r	r	r	d	d	d	d	d	b	b	b

rrr / bbb - register field:

000 - GReg[0]

001 - GReg[1]

...

111 - GReg[7]

dddd - displacement value:

00000 - 0

00001 - 1

...

11111 - 31

### 41.5.2.28 LDF (Load Register from Functional Unit)

#### Operation:

GReg[r] ← [fu\_address]

if (transfer\_error)

SF ← 1

else

SF ← 0

fu\_address is an 8-bit field and depends on addressed functional unit

#### Assembler:

Syntax: ldf r, fu\_address

Example: ldf 0, 13

Loads data coming from the Burst DMA register MD into GReg[0]; it is a 32-bit access with no prefetch

CPU Flags: SF

Cycles: 1+n where n is the number of wait-states that may be inserted by the functional unit

## Instruction Set

**Description:** Sends an 8-bit address on the Functional Unit Bus (FU bus) and stores the data received from the bus in the destination General Register r. If an error occurs during the transfer, the flag SF is set, else it is cleared.

### Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	0	r	r	r	f	f	f	f	f	f	f	f

See the following sections for more details of the LDF instruction usage with each functional unit:

- [Burst DMA Read \(ldf\)](#) for Burst DMA
- [Peripheral DMA Read \(ldf\)-Read Mode](#) for Peripheral DMA

### Instruction Fields:

#### rrr - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

#### ffffff - functional unit source register and action (unspecified values are reserved):

00000000 - MSA

00000100 - MDA

00001001 - MD byte

00001010 - MD halfword

00001011 - MD word

00001100 - MS

00101001 - MD byte - prefetch

00101010 - MD halfword - prefetch

00101011 - MD word - prefetch

01000000 - DSA

11000000 - PSA  
 11001000 - PD  
 11010000 - PDA  
 11011000 - PD in copy mode (rrr contents are lost)  
 11101000 - PD - prefetch next data  
 11111111 - PS

### 41.5.2.29 LDI (Load Register with Immediate Value)

#### Operation:

$\text{GReg}[r] \leftarrow \text{immediate}$

#### Assembler:

Syntax: `ldi r,immediate`

Example: `ldi 6,1`

loads decimal value 1 into GReg[6]

CPU Flags: Unaffected

Cycles: 1

Description: Stores a 0-extended immediate value in a General Register. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	1	r	r	r	i	i	i	i	i	i	i	i

#### Instruction Fields:

##### rrr - register field:

000 - GReg[0]  
 001 - GReg[1]  
 010 - GReg[2]  
 011 - GReg[3]  
 100 - GReg[4]  
 101 - GReg[5]

## Instruction Set

110 - GReg[6]

111 - GReg[7]

iiiiiii - immediate value:

00000000 - 0

00000001 - 1

...

11111110 - 254

11111111 - 255

### 41.5.2.30 LDRPC (Load from RPC to Register)

#### Operation:

GReg[r] ← RPC

#### Assembler:

Syntax: ldrpc r

Example: ldrpc 3

copies RPC to GReg[3]

CPU Flags: Unaffected

Cycles: 1

Description: Stores the contents of the RPC in a General Register. That instruction may be used to have more than one level of subroutines.

#### Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	0	1	0	1	0

#### Instruction Fields:

rrr - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

```

101 - GReg[5]
110 - GReg[6]
111 - GReg[7]

```

### 41.5.2.31 LOOP (Hardware Loop)

#### Operation:

```

if (ff%2 == 0)
    SF ← 0
if (ff/2 == 0)
    DF ← 0
if ((GReg[0] == 0) || (SF == 1) || (DF == 1))
    PC ← PC + loop_size + 1
else
{
    SPC ← PC + 1
    EPC ← PC + loop_size + 1
    LM ← 1
    PC ← PC + 1
}

```

during every instruction execution in the loop:

```

if ((SF == 1) || (DF == 1))
{
    LM ← 0
    PC ← EPC
}
else if ((PC + 1) == EPC)
{
    GReg[0] ← GReg[0] - 1
    if (GReg[0] == 0)
    {
        LM ← 0
        PC ← EPC
    }
}

```

## Instruction Set

```
    }  
    else  
        PC ← SPC  
    }  
else  
    PC ← nextPC(instruction)
```

after the execution of the last instruction of the loop body:

```
if (GReg[0] == 0)  
    T ← 1  
else  
    T ← 0
```

### Assembler:

Syntax: `loop n{,ff}`

Example: `loop 3,1`

Executes GReg[0] times the instructions comprised between PC+1 and PC+3 (included); ff=1 clears the DF flag before starting the loop. When omitted, the ff field is set to 0 (clearing both SF and DF).

CPU Flags: LM[1:0], T

Cycles: 2 when the loop count (GReg[0]) is 0 or SF or DF is set at loop start, 1+1 when the loop starts but exits abnormally (SF or DF set inside the loop which adds 1 cycle to the offending load or store to jump to EPC), 1 when the loop is executed normally

Description: The loop instruction executes a sequence of instructions several times. The number of times is given by the contents of GReg[0], the loop counter. SDMA will jump to the first instruction after the end of the loop if the value in GReg[0] is 0. Otherwise the SDMA enters loop mode. It sets the most significant bit of the LM flag that will only be reset once the last instruction of the last loop is executed. The instructions in the loop are executed GReg[0] times. The management of fault flags (SF and DF) is as follows. When entering the hardware loop, SF and DF can be cleared according to the ff field of the instruction. After that operation, if any flag is still set the loop will not be executed. The SDMA will jump to the first instruction after the end of the loop without entering loop mode. During the execution of the loop, if any fault flag is set by a LD, LDF, ST, or STF instruction, the SDMA will immediately exit loop mode and jump to the first instruction after the end of the loop. In that case, GReg0 is not decremented for that last piece of the loop body execution (even if the SF or DF flag is set at the last instruction of the loop body). The T flag reflects the state of GReg[0] after the end of the loop, which is an indicator of the complete execution of the loop. If the loop exited because of an error (SF

or DF set), GReg[0] will not be 0 at the end of the loop, hence T will be cleared. If the loop executes without fault, GReg[0] will be 0 at the end of the loop, hence T will be set. The boundary case when a source or destination fault occurs at the last instruction of the last loop is considered as an anticipated exit of the loop, which causes the T flag to be cleared. If the last instruction executed before leaving the hardware loop also tries to modify the T flag, the flag is updated according to the value of GReg[0], NOT according to the result of the last executed instruction.

#### Limitations:

1. 1. Jump instructions (JMP, JMPR, JSR, JSRR, BF, BT, BSF, BDF) are not allowed inside the hardware loop.
2. 2. GReg[0] cannot be written to inside the hardware loop (it can be read).
3. 3. The empty loop (0 instruction in the body) is forbidden.
4. 4. If GReg[0] == 0 at the start of the loop, which causes a jump to EPC, the T flag is not updated.

#### Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	1	1	0	f	f	n	n	n	n	n	n	n	n

#### Instruction Fields:

ff - flags field:

00 - clear SF and clear DF

01 - clear DF

10 - clear SF

11 - no clear

#### nnnnnnnn - loop size

00000000 - empty loop: forbidden value

00000001 - 1 instruction in the loop

00000010 - 2 instructions in the loop

...

11111111 - 255 instructions in the loop

### 41.5.2.32 LSL1 (Logical Shift Left by 1 Bit)

#### Operation:

## Instruction Set

$\text{GReg}[r] : \{b30, \dots, b1, b0, 0\} \leftarrow \text{GReg}[r] : \{b31, b30, \dots, b1, b0\}$

### Assembler:

Syntax: `lsl1 r`

Example: `lsl1 2`

multiplies by 2 the value in GReg[2]

CPU Flags: Unaffected

Cycles: 1

Description: Shift the bits of any General Register to the left. The right bit (bit 0) is set to 0. No overflow is detected by the hardware.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	1	1	1

### Instruction Fields:

rrr - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

## 41.5.2.33 LSR1 (Logical Shift Right by 1 Bit)

### Operation:

$\text{GReg}[r] : \{0, b31, b30, \dots, b1\} \leftarrow \text{GReg}[r] : \{b31, b30, \dots, b1, b0\}$

### Assembler:

Syntax: `lsr1 r`

Example: `lsr1 4`

divides by 2 the unsigned value contained in GReg[4]



CPU Flags: Unaffected

Cycles: 1

Description: Shift the bits of any General Register to the right. The left bit (bit 31) is set to 0.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	1	0	1

Instruction Fields:

rrr - destination register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

### 41.5.2.34 MOV (Logical Move)

**Operation:**

$\text{GReg}[r] \leftarrow \text{GReg}[s]$

**Assembler:**

Syntax: `mov r,s`

Example: `mov 4,0`

copies GReg[0] to GReg[4]

CPU Flags: Unaffected

Cycles: 1

Description: Move the contents of the source General Register *s* to the destination General Register *r*.

Instruction Format

## Instruction Set

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	0	0	1	s	s	s

## Instruction Fields:

### rrr / sss - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

## 41.5.2.35 NOTIFY (Notify to ARM platform)

### Operation:

```
if (jjj & 4 == 0)
{
    if (jjj&2 == 2)
        HE[CCR] ← 0
    if (jjj&1== 1)
        HI[CCR] ← 1
}
else if (jjj == 4)
    EP[CCR] ← 0
else
```

(CCR stands for Current Channel Register)

### Assembler:

Syntax: notify jjj

Example: notify 3

clears the HE bit for the current channel and sends an interrupt to the Host for the current channel

CPU Flags: Unaffected

Cycles: 1

Description: Clears one of the channel enabling bits (HE or EP for the corresponding channel number) if required, sends an interrupt to the corresponding ARM platform by setting the appropriate flag if required (HI for the corresponding channel number).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	j	j	j	0	0	0	0	0	0	0	1

jjj - Channel Flags field:

000 - unused

001 - set HI for the current channel

010 - clear HE for the current channel

011 - clear HE, set HI for the current channel

100 - clear EP for the current channel

101 - RESERVED

110 - RESERVED

111 - RESERVED

## 41.5.2.36 OR (Logical OR)

**Operation:**

$\text{GReg}[r] \leftarrow \text{GReg}[s] \mid \text{GReg}[r]$

**Assembler:**

Syntax: `or r,s`

Example: `or 3,6`

ORs GReg[3] and GReg[6] and stores the result in GReg[3]

CPU Flags: Unaffected

Cycles: 1

## Instruction Set

**Description:** Performs the OR of the source General Register s and the destination General Register r, and stores the result in the destination General Register r.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	1	0	1	s	s	s

**Instruction Fields:**

rrr / sss - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

### 41.5.2.37 ORI (Logical OR with Immediate Value)

**Operation:**

$\text{GReg}[r] \leftarrow \text{GReg}[r] \mid \text{immediate}$

**Assembler:**

Syntax: `ori r,immediate`

Example: `ori 1,56`

ORs GReg[1] and the decimal value 56 and stores the result in GReg[1]

CPU Flags: unaffected

Cycles: 1

**Description:** Performs an OR between a 0-extended 8-bit immediate value and a General Register; stores the result in the General Register. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	1	r	r	r	i	i	i	i	i	i	i	i

## Instruction Fields:

### rrr - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

### iiiiiii - immediate value:

00000000 - 0

00000001 - 1

...

11111110 - 254

11111111 - 255

## 41.5.2.38 RET (Return from Subroutine)

### Operation:

PC  $\leftarrow$  RPC

### Assembler:

Syntax: ret

CPU Flags: Unaffected

Cycles: 2

Description: Return from subroutine.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

### 41.5.2.39 REVB (Reverse Byte Order)

#### Operation:

$$GReg[r] : \{B3, B2, B1, B0\} \leftarrow GReg[r] : \{B0, B1, B2, B3\}$$

#### Assembler:

Syntax: `revb r`

Example: `revb 5`

reverses bytes order in GReg[5]

CPU Flags: Unaffected

Cycles: 1

Description: Reverse the byte order of any General Register.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	0	0	0

#### Instruction Fields:

rrr - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

### 41.5.2.40 Reverse Low Order Bytes(REVBLO)

#### Operation:

$$GReg[r] : \{B3, B2, B0, B1\} \leftarrow GReg[r] : \{B3, B2, B1, B0\}$$

**Assembler:**

Syntax: revblo r

Example: revblo 0

reverses low order bytes in GReg[0]

CPU Flags: Unaffected

Cycles: 1

Description: Reverse both low order bytes of any General Register.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	0	0	1

**Instruction Fields:**

rrr - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

**41.5.2.41 ROR1 (Rotate Right by 1 Bit)****Operation:**

$$\text{GReg}[r] : \{b0, b31, b30, \dots, b1\} \leftarrow \text{GReg}[r] : \{b31, b30, \dots, b1, b0\}$$
**Assembler:**

Syntax: ror1 r

Example: ror1 3

rotates bits to the right in GReg[3]

CPU Flags: Unaffected

## Instruction Set

Cycles: 1

Description: Rotate the bits of any General Register to the right.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	1	0	0

Instruction Fields:

rrr - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

## 41.5.2.42 RORB (Rotate Right by 1 Byte)

**Operation:**

$\text{GReg}[r] : \{B0, B3, B2, B1\} \leftarrow \text{GReg}[r] : \{B3, B2, B1, B0\}$

**Assembler:**

Syntax: `rorb r`

Example: `rorb 2`

rotates bytes to the right in GReg[2]

CPU Flags: Unaffected

Cycles: 1

Description: Rotate the bytes of any General Register to the right.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	0	0	0	1	0	0	1	0



## Instruction Fields:

### rrr - register field:

000 - GReg[0]  
 001 - GReg[1]  
 010 - GReg[2]  
 011 - GReg[3]  
 100 - GReg[4]  
 101 - GReg[5]  
 110 - GReg[6]  
 111 - GReg[7]

## 41.5.2.43 SOFTBKPT (Software Breakpoint)

### Operation:

Stops the current script and enters debug mode

### Assembler:

```
softbkpt
```

CPU Flags: Unaffected

Description: When the core executes this instruction, it has the same effect as receiving a debug request from the OnCE or via the external debug request input: the script execution halts, the PCU enters its debug state and waits for the OnCE commands that are described in [OnCE and Real-Time Debug](#).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1

## 41.5.2.44 ST (Store Register)

### Operation:

```
[GReg[b] + displacement] ← GReg[r]
```

```
if (transfer_error)
```

## Instruction Set

DF ← 1

else

DF ← 0

## Assembler:

Syntax: st r, (b, displacement)

Example: st 7, (0, 9)

stores the value from GReg[7] into memory at address obtained by adding decimal value 9 to GReg[0]

CPU Flags: DF

Cycles: 2+n where n is 0 for ROM, RAM or memory mapped registers, and n is the number of wait-states of the peripheral for a peripheral access

Description: Adds a 5-bit 0-extended displacement to a base address in General Register b; the result is the address of the data to store on the DM bus. The data sent on the bus comes from the source General Register r. If an error occurs during the transfer, the flag DF is set, else it is cleared.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	1	1	r	r	r	d	d	d	d	d	b	b	b

## Instruction Fields:

rrr / bbb - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

dddd - displacement value:

00000 - 0

00001 - 1

...

11111 - 31

### 41.5.2.45 STF (Store Register in Functional Unit)

#### Operation:

```
[fu_address] ← GReg[r] 0
```

```
if (transfer_error) 0
```

```
DF ← 1 0
```

```
else 0
```

```
DF ← 0
```

fu\_address is an 8-bit field

#### Assembler:

Syntax: `stf r, fu_address`

Example: `stf 3, 0x2B`

stores the 32-bit contents of GReg[3] to the Burst DMA register MD; waits until the flush to external memory is completed

CPU Flags: DF

Cycles: 1+n where n is the number of wait-states that may be inserted by the functional unit

Description: Sends an 8-bit address on the Functional Unit Bus (FU bus) and sends the contents of the source General Register r on the bus. If an error occurs during the transfer, the flag DF is set, else it is cleared.

**Table 41-47. Instruction Format**

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	1	0	1	r	r	r	f	f	f	f	f	f	f	f

See the following sections for more details of the STF instruction usage with each functional unit:

- [Burst DMA Write \(stf\)](#) for Burst DMA
- [Peripheral DMA Write \(stf\)-Write Mode](#) for Peripheral DMA

Instruction Fields:

rrr - register field:

## Instruction Set

000 - GReg [0]

001 - GReg [1]

010 - GReg [2]

011 - GReg [3]

100 - GReg [4]

101 - GReg [5]

110 - GReg [6]

111 - GReg [7]

ffffff - functional unit destination register and action (unspecified values are reserved):

00000000 - MSA in incremented mode

00000100 - MDA in incremented mode

00001001 - MD byte

00001010 - MD halfword

00001011 - MD word

00001100 - clear MS error flag

00001111 - MS

00010000 - MSA in frozen mode

00010100 - MDA in frozen mode

00011000 - MD in copy mode - number of words in rrr

00100000 - MSA in incremented mode - start prefetch

00101000 - MD no data - flush

00101001 - MD byte - flush

00101010 - MD halfword - flush

00101011 - MD word - flush

00110000 - MSA in frozen mode - start prefetch

11000001 - PSA in frozen mode - 8-bit data width  
11000010 - PSA in frozen mode - 16-bit data width  
11000011 - PSA in frozen mode - 32-bit data width  
11000101 - PSA in incremented mode - 8-bit data width  
11000110 - PSA in incremented mode - 16-bit data width  
11000111 - PSA in incremented mode - 32-bit data width  
11001000 - PD  
11001001 - PSA in decremented mode - 8-bit data width  
11001010 - PSA in decremented mode - 16-bit data width  
11001011 - PSA in decremented mode - 32-bit data width  
11001100 - clear PS error flag  
11001101 - PSA data width becomes 8-bit  
11001110 - PSA data width becomes 16-bit  
11001111 - PSA data width becomes 32-bit  
11010001 - PDA in frozen mode - 8-bit data width  
11010010 - PDA in frozen mode - 16-bit data width  
11010011 - PDA in frozen mode - 32-bit data width  
11010101 - PDA in incremented mode - 8-bit data width  
11010110 - PDA in incremented mode - 16-bit data width  
11010111 - PDA in incremented mode - 32-bit data width  
11011001 - PDA in decremented mode - 8-bit data width  
11011010 - PDA in decremented mode - 16-bit data width  
11011011 - PDA in decremented mode - 32-bit data width  
11011101 - PDA data width becomes 8-bit  
11011110 - PDA data width becomes 16-bit  
11011111 - PDA data width becomes 32-bit  
11100001 - PSA in frozen mode - 8-bit data width - prefetch data

11100010 - PSA in frozen mode - 16-bit data width - prefetch data  
 11100011 - PSA in frozen mode - 32-bit data width - prefetch data  
 11100101 - PSA in incremented mode - 8-bit data width - prefetch data  
 11100110 - PSA in incremented mode - 16-bit data width - prefetch data  
 11100111 - PSA in incremented mode - 32-bit data width - prefetch data  
 11101001 - PSA in decremented mode - 8-bit data width - prefetch data  
 11101010 - PSA in decremented mode - 16-bit data width - prefetch data  
 11101011 - PSA in decremented mode - 32-bit data width - prefetch data  
 11101101 - PSA data width becomes 8-bit - prefetch data  
 11101110 - PSA data width becomes 16-bit - prefetch data  
 11101111 - PSA data width becomes 32-bit - prefetch data  
 11111111- PS

## 41.5.2.46 SUB (Subtract)

### Operation:

$\text{GReg}[r] \leftarrow \text{GReg}[r] - \text{GReg}[s]$

$T \leftarrow (\text{GReg}[r] == 0)$

### Assembler:

Syntax: `sub r,s`

Example: `sub 4,7`

SUBtracts GReg[7] from GReg[4] and stores the result in GReg[4]

CPU Flags: T

Cycles: 1

Description: Subtracts the source General Register s from the destination General Register r, and stores the result in the destination General Register r. The T flag is set if the result of the operation is 0; it is cleared if the result is not 0.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	1	0	0	s	s	s

## Instruction Fields:

rrr / sss - register fields:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

### 41.5.2.47 SUBI (Subtract with Immediate)

#### Operation:

$\text{GReg}[r] \leftarrow \text{GReg}[r] - \text{immediate}$

$T \leftarrow (\text{GReg}[r] == 0)$

#### Assembler:

Syntax: `sub r,immediate`

Example: `sub 1,255`

SUBtracts decimal value 255 from GReg[1] and stores the result in GReg[1]

CPU Flags: T

Cycles: 1

Description: Subtracts a 0-extended 8-bit immediate value from a General Register; stores the result in the General Register. The flag T is set when the result of the operation is 0; otherwise, it is cleared. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	1	0	0	r	r	r	i	i	i	i	i	i	i	i

## Instruction Fields:

## Instruction Set

rrr - register field:

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

iiiiiii - immediate value:

00000000 - 0

00000001 - 1

...

11111110 - 254

11111111 - 255

### 41.5.2.48 TST (Test with Zero)

#### Operation:

$T \leftarrow ((\text{GReg}[s] \ \& \ \text{GReg}[r]) \neq 0)$

#### Assembler:

Syntax: `tst r,s`

Example: `tst 2,3`

ANDs GReg[2] and GReg[3] and sets T if the result is non-null

CPU Flags: T

Cycles: 1

Description: Performs the AND of the source General Register s and the destination General Register r, and sets T if the result is not 0, clears T if the result is 0.

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	1	0	0	0	s	s	s



**Instruction Fields:****rrr / sss - register field:**

000 - GReg[0]

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

**41.5.2.49 TSTI (Test Immediate)****Operation:** $T \leftarrow ((\text{GReg}[r] \ \& \ \text{immediate}) \neq 0)$ **Assembler:**Syntax: `tsti r,immediate`Example: `tsti 5,13`

ANDs GReg[5] and decimal value 13 and sets T if the result is non-null

CPU Flags: T

Cycles: 1

Description: Performs the AND of a 0-extended 8-bit immediate value and the destination General Register r, and sets T if the result is not 0, clears T if the result is 0. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	1	0	0	0	r	r	r	i	i	i	i	i	i	i	i

**Instruction Fields:****rrr - destination register field:**

000 - GReg[0]

## Instruction Set

001 - GReg[1]

010 - GReg[2]

011 - GReg[3]

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

iiiiiii - immediate value:

00000000 - 0

00000001 - 1

...

11111110 - 254

11111111 - 255

## 41.5.2.50 XOR (Logical Exclusive OR)

### Operation:

$\text{GReg}[r] \leftarrow \text{GReg}[s] \wedge \text{GReg}[r]$

### Assembler:

Syntax: `xor r,s`

Example: `xor 0,3`

XORs GReg[0] and GReg[3] and stores the result in GReg[0]

CPU Flags: Unaffected

Cycles: 1

Description: Performs the eXclusive OR of the source General Register s and the destination General Register r, and stores the result in the destination General Register r.

### Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	0	0	r	r	r	1	0	0	1	0	s	s	s

### Instruction Fields:

rrr / sss - register field:

000 - GReg[0]  
 001 - GReg[1]  
 010 - GReg[2]  
 011 - GReg[3]  
 100 - GReg[4]  
 101 - GReg[5]  
 110 - GReg[6]  
 111 - GReg[7]

### 41.5.2.51 XORI (Exclusive OR with Immediate)

#### Operation:

$\text{GReg}[r] \leftarrow \text{GReg}[r] \wedge \text{immediate}$

#### Assembler:

Syntax: `xori r,immediate`

Example: `xor 7,5`

XORs GReg[5] and decimal value 5 and stores the result in GReg[7]

CPU Flags: Unaffected

Cycles: 1

Description: Performs an eXclusive OR between a 0-extended 8-bit immediate value and a General Register; stores the result in the General Register. The immediate value is the low-order byte of the instruction and has a maximum value of 255 (0xFF).

Instruction Format

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
0	0	0	1	0	r	r	r	i	i	i	i	i	i	i	i

#### Instruction Fields:

rrr - register field:

000 - GReg[0]  
 001 - GReg[1]  
 010 - GReg[2]  
 011 - GReg[3]

## Software Restrictions

100 - GReg[4]

101 - GReg[5]

110 - GReg[6]

111 - GReg[7]

iiiiiii - immediate value:

00000000 - 0

00000001 - 1

...

11111110 - 254

11111111 - 255

### 41.5.2.52 YIELD, YIELDGE (DONE, Yield)

By default, unsupported assembler syntax. Can be aliased to the corresponding done instructions (yield = done 0; yieldge = done 1). Refer to the done instruction description [DONE \(DONE, Yield\)](#).

## 41.6 Software Restrictions

### 41.6.1 Unsupported Burst DMA Access Sequence

The SDMA does not support triggering a pre-fetch followed by a flush of the Burst DMA without reading or writing any data. If the flush occurs while the background pre-fetch DMA operation is still in progress, it could result in un-defined behavior.

An example of the sequence which could result in undefined results is shown in the following example:

Instruction sequence not supported

```
stf r1, MSA|PF      ; Update source address, triggers data pre-fetch in the
                    ; background
mov R0,R0            ; Execute multiple assembly instructions, none of which
                    ; read
mov R0,R0            ; or write data to/from MD
stf MD|SZ0|FL        ; Flush FIFO without writing data. If the pre-fetch is still
                    ; in progress when this instruction is executed, there
                    ; could be undefined operation
```

A work-around to avoid any undesirable results is to first read MD to ensure the pre-fetch is complete before the flush is attempted.

Work-Around to previous example

```

stf r1, MSA|PF      ; Update source address, triggers data pre-fetch.
mov R0,R0            ; Execute multiple assembly instructions, none of which
                    ; read
mov R0,R0            ; or write data to/from MD
ldf r2, MD           ; dummy read of MD to ensure pre-fetch is complete
                    ; before the next instruction
stf MD|SZ0|FL        ; Flush FIFO without writing data

```

## 41.7 Application Notes

### 41.7.1 Data Structures for Boot Code and Channel Scripts

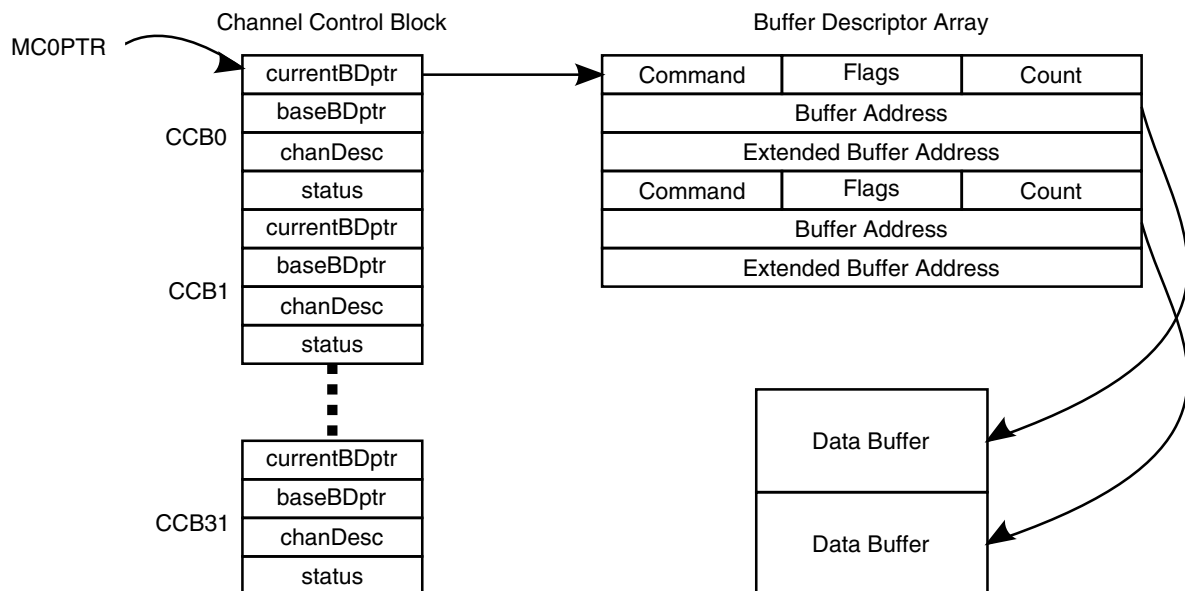
SDMA boot code downloads the different channel contexts and the scripts that will be executed on SDMA channels during the application.

The boot code is run after reset when channel 0 is started by the ARM platform. The boot code is also known as channel 0 script.

The boot code is based on the Channel Control Block (CCB) and Buffer Descriptor (BD) mechanisms that are data structures located into the ARM platform memory space. With these data structures, it is possible to instruct SDMA to download scripts and contexts but also to dump a context or a script to a destination data buffer. Channel scripts also use the CCB and BD data structures to pass instructions and/or pointers to data to be copied.

The format, processing, and field definition of the CCB and BD are defined and performed entirely by the software script rather than the SDMA hardware. An overview of the format and structure is provided here, but for complete details refer to the SDMA software documentation (see [SDMA Scripts](#)).

The CCB and BD data structures are accessed by SDMA using DMA and processed by the SDMA scripts. The ROM contains common sub-routines for processing these data structures which may be called by the bootload and channel scripts.



**Figure 41-16. Data Structures Layout**

The previous figure shows an example how these data structures are linked to pass command and pointers to data buffers. The SDMA's MC0PTR register holds the base address of the Channel 0 Control Block (CCB0). The Channel 0 control block holds a pointer to the array of buffer descriptors. The buffer descriptors are used to tell the channel 0 (boot channel) what to do as described [Buffer Descriptor Format](#).

### 41.7.1.1 Buffer Descriptor Format

Buffer descriptors are three longs (32-bit words) in size as, shown in the figure found [here](#).

A buffer descriptor describes the properties of the data buffer it points to. The buffer descriptors can be used for linear or circular data buffers in the ARM platform processor memory. The CCB contains a pointer to the base BD as well as the current BD.

**Table 41-48. Buffer Descriptor**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Command								-	-	L	R	I	C	W	D	Count																
Buffer Address																																
Extended Buffer Address																																

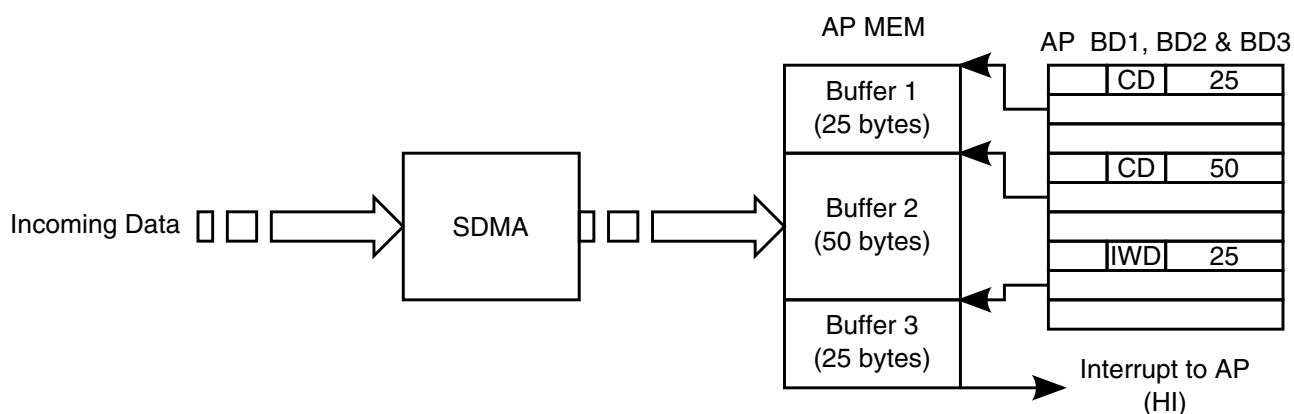
**Table 41-49. Buffer Descriptor Field Descriptions**

Field	Description
31-24 Command	Command. The command field is used to differentiate operations performed within a script when the script accesses this particular buffer descriptor. The use of this field can be defined by the script. The command values defined for the bootstrap script are defined in <a href="#">Buffer Descriptor Commands for Bootstrap scripts</a> . Refer to the individual script definition in script library documents in <a href="#">SDMA Scripts</a> for command field definitions for other scripts.
23	Reserved
22	Reserved
21 L	Last Buffer Descriptor: This bit is set in SDMA IPC scripts to indicate to the receiving Core that the transfer has ended. Whenever the source finishes transferring the count it wanted to transfer, it sets LAST_BIT in the destination BD, to let the destination know that transfer is over. This bit also tells the destination software that when it processes the destination BDs, they need not process any BD after the BD with the LAST_BIT set. For example, when the DSP prepares a single buffer descriptor with count equals to 25 and ARM platform prepares a single buffer descriptor with count equals 100. When 25 bytes have been transferred from DSP to ARM platform, the DSP buffer descriptor is normally closed while the ARM platform buffer descriptor will have the L bit set and the byte count updated to 25.
20 R	erroR. Indicates an error occurred on the channel's buffer descriptor requested command. Some scripts may overwrite the command field with an error code indicating the source of the error. 0 No Error 1 Error
19 I	Interrupt. When SDMA has finished to process data transfer attached to this buffer descriptor, send an interrupt to the ARM platform. 0 No Interrupt 1 Interrupt the processor when BD is complete
18 C	CONTinuous. This buffer is allowed to receive multiple transmit buffers or is allowed to transmit to multiple receive buffers. The Continuous bit is decoded at the end of the processing of a BD to determine if the SDMA script must open a new BD to potentially continue the data transfer. 0 No further buffer descriptors 1 SDMA should move to the next Buffer descriptor after this one
17 W	Wrap. Indicates if this buffer descriptor is the last one for the channel control block. When encountering this bit set, the SDMA scripts updates the CurrentBD pointer to point to the first Buffer Descriptor of the array. This bit is set if the ARM platform wants to organize the array of BD in a circular way (like a ring). When all BD have been processed and if Wrap bit and CONTinuous bit are set in the last BD, the SDMA script will wrap around and it will try to re-open the first BD. 0 No Error 1 Wrap to first buffer descriptor after this one is processed.
16 D	D - "Done": bit 16: indicates the "ownership" of the buffer descriptor. When D=0 the host owns the buffer descriptor; when D=1 SDMA owns the buffer descriptor. In the case of the channel 0, D=1 indicates the SDMA has not yet processed this buffer, D=0 indicates the SDMA has processed this buffer. 0 ARM platform owns the buffer. 1 SDMA owns the buffer
15-0 Count	Count. the count field (bit 15-0) indicates the size of the data to be transmitted, the size of the data buffer pointed to by the buffer descriptor. The SDMA memory structure is different for program memory (16-bits shorts/half-words) and data memory (32-bits long). For channel 0 buffer descriptors, Count is expressed in 16-bit half-words when PM is addressed and in 32-bit words when DM is addressed. Count is typically expressed in bytes for other channel scripts, but the unit is dependant on the script.
31-0	Buffer address. Address pointer to the data buffer.
31-0	Extended buffer address. Additional pointer or other information required by some scripts.

The buffer descriptors form an array of programmable size. If the last buffer descriptor is marked by the Wrap flag-bit  $W=1$ , the array of buffer descriptor is treated as a ring with some logically continuous portion owned by the ARM platform with  $D=0$ , and the remainder owned by the SDMA with  $D=1$ . The count field of the buffer descriptor indicates how much data has been transmitted.

If ARM platform has prepared 3 buffers to be filled by the SDMA script, it has also prepared 3 BD, one for each buffer. The *Cont* and *Wrap* bits are used to organize the buffers in a circular way. For example, *CONTInous* bit is set to 1 in the 2 first BDs and *Wrap* is set in the 3<sup>rd</sup> BD. The SDMA script opens and processes BD#1. Since *CONTInous* bit is set for this BD, the SDMA will open the second BD and it will process it. Each time a BD is processed, its *Done* bit is reset by the SDMA. After the 3<sup>rd</sup> BD, if *CONTInous* is not set but if *Wrap* is set, the SDMA script stops here and the next time the channel will be triggered, the script will open the BD pointed by the currentBDptr pointer of the CCB and it will correspond to the first buffer descriptor.

If the *CONTInous* bit and *Wrap* bits are both set in the 3<sup>rd</sup> BD, the script will close it and it will try to open the first BD. An error may occur at this point if the BD#1 has already been processed and its *Done* bit is 0. The SDMA script cannot process a BD with a *Done* bit to 0. It means the BD is not ready to be processed. To avoid this situation, the *CONTInous* bit should not be set for the last BD if *Wrap* is set, and the Interrupt flag must set for the last BD. It will warn the owner of the BD that all the BDs have been processed and it has to re-set to 1 the *Done* bit of all the BD's if it desires the SDMA to fill them again. Basically, if the ARM platform expects the SDMA to fill up the buffers in a circular fashion, then it's the responsibility of the ARM platform to set the *Done* bit of a buffer descriptor at an appropriate time.



**Figure 41-17. Buffer Descriptor Flow**



The previous figure shows an example buffer descriptor flow. When the incoming data is stored and fills the first buffer of 25 bytes, the SDMA script opens the second BD because the CONTinuous bit was set. Then next incoming data is put in the second buffer. After receiving 50 bytes, the second buffer descriptor is also closed. The Done bit is reset and the third BD is opened. After receiving another 25 bytes, the third buffer is full and an interrupt is sent to the ARM platform because the Interrupt flag is set in the 3rd BD. The CONTinuous flag is not present the transfer is over. The next time the script will be triggered, the BD to be opened will be the first buffer descriptor since the Wrap flag was set in the 3rd BD. It is the ARM platform responsibility to set the Done bit of all the BD if it wants to use the same buffers.

### 41.7.1.2 Buffer Descriptor Commands for Bootload scripts

The command field of the buffer descriptor is defined separately for each script.

The following table lists the buffer descriptor commands defined for the channel 0 bootload script.

**Table 41-50. Channel Zero Buffer Descriptor Commands**

Command Field (binary)	Command	Description	Buffer Address	Extended Buffer Address
0000_0001 (0x01)	C0_SET_DM	Load SDMA data memory (RAM) from ARM platform memory buffer	ARM platform memory source address	SDMA memory destination address
0000_0010 (0x02)	C0_GET_DM	Copy SDMA data memory (RAM) to ARM platform memory buffer	ARM platform memory destination address	SDMA memory source address
0000_0100 (0x04)	C0_SET_PM	Load SDMA program memory (RAM) from ARM platform memory buffer	ARM platform memory source address	SDMA memory destination address
0000_0110 (0x06)	C0_GET_PM	Copy SDMA program memory (RAM) to ARM platform memory buffer	ARM platform memory destination address	SDMA memory source address
cccc_c111 (0x07   CHN)	C0_SETCTX	Load Context for channel cccc into SDMA RAM from ARM platform memory buffer	ARM Platform memory source address	-
cccc_c011 (0x03   CHN)	C0_GETCTXT	Copy Context for channel ccccc from SDMA RAM to ARM platform memory buffer	ARM platform memory destination address	-

The Channel 0 bootload commands are summarized as follows:

- **C0\_SET\_[PM-DM]:** load the buffer descriptor data in the SDMA local memory at the address pointed to by the "extended buffer address" field. The SDMA RAM can be seen as a Program Memory (PM, 16-bit address) or Data Memory (DM 32-bit address). When C0\_SET\_PM is used, the count field is expressed in "shorts" (16-bit

half words), this command can be used to download scripts. When C0\_SET\_DM is used, the count field is expressed in "long" (32-bit words), this command can be used to download channel contexts to the context channel area in RAM.

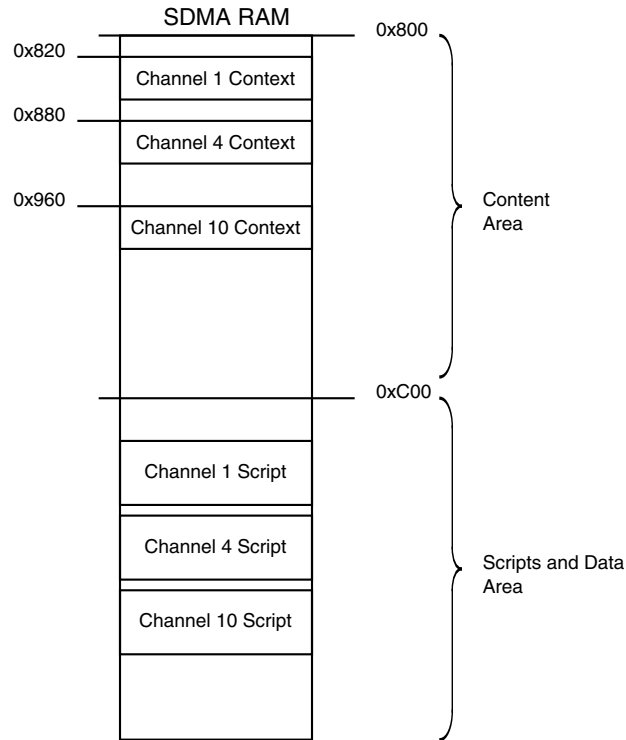
- C0\_GET\_[PM-DM]: write to the buffer descriptor's data buffer the content of the SDMA local memory from the address pointed to by the "extended buffer address" field for the length defined by the count in the buffer descriptor. C0\_GET\_PM is used to dump some part of the Program Memory (may be used to dump context of a channel), therefore count is expressed in "shorts"; while C0\_GET\_DM is used to dump to the buffer descriptor's data buffer, so the count field is in "longs."
- C0\_SETCTX: load a context into the SDMA context page area. The handling script decodes the channel number from the 5 MSB of the command field of the buffer descriptor. Using the channel number the script computes the offset of the context data pointer for the channel relative to the context page base to use as the destination address in SDMA memory. Then the C0\_SET\_DM command explained above is invoked to load SDMA RAM from memory. The counter indicates the size in words of the context structure.
- Command value: (in binary) cccc c111, where ccccc is the channel number (5 bits). For instance, 0x0F means set context for channel 1, 0xFF means set context for channel 31.
- C0\_GETCTX: write to the buffer descriptor's data buffer the content of the SDMA context page area. The handling script decodes the channel number from the 5 MSB of the command field of the buffer descriptor. Using this channel number, the script computes the offset of the context data pointer for the channel relative to the context page base to use as the source address for the copy. Then the C0\_GET\_DM command explained above is invoked to copy the context to memory. The counter indicates the size in words of the context structure.
- Command value: (in binary): cccc c011, where ccccc is the channel number (5 bits). For instance, 0x03 means get context of channel 1, 0xFB means get context of channel 31.

### NOTE

To download channel context, C0\_SETDM and C0\_SETCTXT command can be used but the second one is easier because the channel number is embedded into the command field, whereas with the C0\_SETDM, the pointer to the channel context area must be written into the extended buffer address field of the buffer descriptor.

### 41.7.1.3 Example of Buffer Descriptors for Channel 0.

Figure 41-19 illustrates the buffer descriptors that must be set in ARM platform memory space, before execution of boot code, to download contexts and scripts of channels 1, 4, and 10. After boot code execution, SDMA memory will be populated with the different contexts and scripts as presented in the following figure.



**Figure 41-18. Example of SDMA RAM After Boot Session**

SDMA Register

MCOPTR

Channel Control Block

CurrentBDptr
BaseBDptr
chanDesc
status

Channel 0 Buffer Descriptor Array

31	24	23	22	21	20	19	18	17	16	15	0
00001111					0	0	1	0	1		20
Buffer Address											
Extended Buffer Address (Unused)											
00100111					0	0	1	0	1		20
Buffer Address											
Extended Buffer Address (Unused)											
01010111					0	0	1	0	1		20
Buffer Address											
Extended Buffer Address (Unused)											
00000100					0	0	1	0	1		10
Buffer Address											
Extended Buffer Address											
00000100					0	0	1	0	1		40
Buffer Address											
Extended Buffer Address											
00000100					0	1	0	0	1		50
Buffer Address											
Extended Buffer Address											

BD1 - SET CONTEXT CH#1  
Interrupt = 0,  
Cont=1, Done = 1

BD2 - SET CONTEXT CH#4  
Interrupt = 0,  
Cont=1, Done = 1

BD3 - SET CONTEXT CH#10  
Interrupt = 0,  
Cont=1, Done = 1

BD4 - SET\_PM  
Interrupt = 0,  
Cont=1, Done = 1

BD5 - SET\_PM  
Interrupt = 0,  
Cont=1, Done = 1

BD6 - SET\_PM  
Interrupt = 1,  
Cont=0, Done = 1

SDMA RAM

Context Area
Scripts Area

AP Memory Space

Channel 1 context (32 longs)
Channel 4 context (32 longs)
Channel 10 context (32 longs)
Channel 1 script (16 shorts)
Channel 4 script (64 shorts)
Channel 10 script (80 shorts)

#### 41.7.1.4 Channel Context

There are 32 channel context memory structures pointed to by the local save area pointer. These channel context memory structures are fixed.

The script in the SDMA computes the memory offset for a given channel based on the structure length and channel number. Figure below shows the structure of the channel context as it is saved in the SDMA local memory (RAM).

A channel context consists in 24 words, one per register. A total of 32 words are reserved for every channel. The additional 8 words are called scratch ram and they are dedicated to each channel. This memory area is commonly used for stack management.

The structure is divided in 4 areas:

- Channel status registers
- General purpose registers
- Functional units state registers reflecting the state of the ARM platform DMAs (Burst and Peripheral DMA).
- Scratch RAM

The details of the channel context status registers are described in the following figure.

The PC field of the first long register must point to the SDMA RAM address where the script that will be executed on the channel is located and this value equals the one stored in the extended buffer address of the buffer descriptor with C0\_SETPM command.

31	30	29	16	15	14	13	0
SF	—	RPC	T	—	PC		
LM		EPC	DF	—	SPC		

SF: Source fault while loading data  
 RPC: Return program counter  
 T: Test bit: status of arithmetic and test instructions  
 PC: Program counter  
 LM: Loop mode  
 EPC: Loop end program counter  
 DF: Destination fault while storing data  
 SPC: Loop Start program counter

**Figure 41-20. SDMA State Registers (ShPC, ShLoop)**

## 41.7.2 Typical Data Transfer Supported by SDMA DMA Units

This section presents a library of SDMA scripts that perform data transfers through the peripheral DMA and the burst DMA units.

The ARM platform memory and peripherals are devices that either the peripheral DMA or the burst DMA can access. The scripts are given for a peripheral DMA whose address registers are programmed in incremented mode when internal memory is involved. See the following table for the summary.

**Table 41-51. Typical Data Transfers Summary**

Data Transfer	Peripheral DMA	Burst DMA	Comments
ARM platform External Memory ↔ ARM platform External Memory		3	Copy mode Script example, see <a href="#">Burst DMA Unit Copy Mode</a> and <a href="#">External Memory to External Memory</a> .
ARM platform Peripheral ↔ ARM platform Peripheral	3		Copy mode if same data path width Script example, see <a href="#">Peripheral to Peripheral Transfer</a> .
ARM platform External Memory ↔ ARM platform Peripheral	3	3	Data transit through SDMA Script example, see <a href="#">Transfer Between Peripheral and External Memory</a> .
ARM platform External Memory ↔ ARM platform Internal Memory		3	Copy mode Script example, see <a href="#">Transfer Between External Memory and Internal Memory</a> .
ARM platform Internal Memory ↔ ARM platform Internal Memory		3	Copy mode Script example, see <a href="#">Internal Memory to Internal Memory</a> .
ARM platform Internal memory ↔ ARM platform Peripheral	3		Data transit through SDMA Script example, see <a href="#">Transfer Between Peripheral and Internal Memory</a> .

### NOTE

These scripts are provided as examples of how to use DMA blocks to perform required data transfers: They are not "official" programs.

### 41.7.2.1 External Memory to External Memory

This section describes the SDMA script that performs data moves in external memory.

For this particular data transfer, only the burst DMA is used. It is programmed in copy mode, so no data transmits through an SDMA general register.

The SDMA core only monitors data transfer status. It is assumed source and destination address values are already present in two SDMA general registers (r1 and r2). For this example, it is also assumed that a 32-bit word-to-move for source-to-destination address is present in r0 and equals 64.

### Data Moves in External Memory

```

1      stf r1,MSA                      // Source address setup
2      stf r2,MDA                      // Destination address setup
3      ldi r0,0x64                     // 64 words must be transferred from MSA to
MDA
4      ldi r1,0x8
MAIN_XFER:
5      cmpbs r0,r1                    // Is r0 >= 0x8
6      bf LAST_XFER                   // If not, jump to last transfer label
7      stf r1,MD|CPY                  // Copy 8 words from MSA to MDA address.
8      subi r0,0x8                    // Decrement counter
9      jmp MAIN_XFER                  // return to main transfer loop
LAST_XFER:
10     stf r0,MD|CPY                  // perform last transfer

```

All instructions are performed in one cycle (jumps excepted). Instruction 7 triggers a copy transfer: A read burst access of 8-word starts, data is staged in MD and then a write burst of 8 words is executed. Instruction 8, 9, 5, and 6 are executed while the burst access is in progress. If this access is not complete when instruction 7 is executed a second time, SDMA stalls on this instruction as long as the previous copy transfer is not over. In this case, the instruction is no longer a one-cycle instruction.

During the main loop (MAIN\_XFER), r1 always equals 8, so burst lengths are 8 words. On the last ldf |CPY instruction (10), r1 equals the remainder of r0 divided by 8; therefore, the length of bursts triggered in copy mode equal r1 value, which is between 1 and 7.

### 41.7.2.2 Peripheral to Peripheral Transfer

For this data transfer, only the peripheral DMA is used.

It is programmed in copy mode, so no data will transmit through the SDMA general register used in the ldf instruction, but the contents of the general register are lost. The SDMA core only monitors the transfer.

### 41.7.2.2.1 Source and Destination Target Have the Same Data Path Width

When the source and destination target have the same data path width, the following is true:

- Source target is a *half-word* (16-bit) peripheral located at address 0x1002.
- Destination is a *half-word* (16-bit) peripheral located at address 0x2006.

It is assumed the address values are already present in two SDMA general registers (r1, r2). The script for a transfer of 10 half-word is as follows:

#### Same Data Path Width for Source and Destination

```
//SETUP SECTION
1      stf r1, PSA|SZ16|F           //r1=0x1002 Source address register setup
2      stf r2, PDA|SZ16|F           //r2=0x2006 Destination address register
setup
3      bdf ERROR_ADDR_SETUP
4      ldi r0,0xa                   //loop counter is 10
//MAIN LOOP TRANSFER
copy_loop:

5      loop 2,0
6      ldf r7,PD|CPY                //Reads 1 half-word from src and writes to
dest.
7      yield
8      bdf ERROR_DURING_XFER
ERROR_ADDR_SETUP:                  //correction of PSA/PDA setup and jumps to main loop transfer
ERROR_DURING_XFER:
//flag error is set,
//PS can be read to know if error occurs during read or write access.
```

If a data transfer must occur between two word peripherals, only the setup section should be updated. The transfer itself is always performed by the hardware loop instruction.

All instructions are executed in one cycle (change of flow excepted). On instruction 6, a single read access is triggered, read data is staged in PD, and a write-to-destination is executed. When the transfers are in progress, the SDMA can execute the next instructions in parallel. If instruction 6, which performs the copy transfer, is executed while the previous access is not over, SDMA is stalled and instruction ldf is a multi-cycle instruction.

### 41.7.2.2.2 Source and Destination Target Have a Different Data Path Width

When the source and destination target have a different data path width, copy mode cannot be used, and any attempt to initiate a copy transfer immediately raises an error, which is stored in the SF flag.

The following example shows the SDMA code that could transfer 10 words from a *word* (32-bit) peripheral to a *half-word* peripheral whose addresses are preliminary and stored in r1 and r2.



## Different Data Path Width for Source and Destination

```
//SETUP SECTION
1      stf r1, PSA|SZ32|F|PF          //r1=0x1000 and prefetch data
2      stf r2, PDA|SZ16|F            //r2=0x2006
3      bdf ERROR_ADDR_SETUP
4      ldi r0,0xa                     //loop counter is 10
//MAIN LOOP TRANSFER
main_loop_xfer_16_16:
5      loop 6,0
6      ldf r7,PD                      //copy 32-bit of PD in r7
7      stf r7,PD                      //store 16 LSB of r7 in PD and a flush is
executed
8      rorb r7
9      rorb r7                        //16 MSB --> 16 LSB
10     stf r7,PD                      //store 16 LSB of r6 in PD and a flush.
11     yield
```

On instruction 1, when the source address register is programmed and a data prefetch is required, a read access is executed. In parallel, the SDMA executes instructions 2 to 5. On instruction 6, the SDMA tries to read data that was fetched by instruction 1. If data is ready, the ldf will be a one cycle instruction; otherwise, the SDMA is stalled as long as the read access is not finished. Then, the 16 LSB of the read data is stored in PD and automatically flushed to the destination peripheral. In parallel, the SDMA executes the rotation instructions (8, 9), and stores the 16 MSB of the read data into PD. If a previous write access is finished, instruction 10 will be a one-cycle instruction.

The main loop transfer may appear inefficient, but due to wait states imposed to the peripheral DMA each time an external access is performed, a software pipeline is in place. During the time needed to flush PD, the SDMA executes the move and rotation operations. SDMA executes instructions in parallel with DMA accesses.

### 41.7.2.3 Transfer Between Peripheral and External Memory

#### 41.7.2.3.1 Peripheral to External Memory Transfer

A transfer from a peripheral to the external memory controller involves the peripheral DMA and the burst DMA.

The code for transferring 100 word from word peripheral to the external memory would be as follows:

#### Peripheral to External Memory Transfer

```
//SETUP SECTION source and destination addresses are already in r1 and r2
1      stf r1, PSA|SZ16|F|PF          //r1=0x1000 and prefetch 32-bit data
2      stf r2, MDA                    //r2=0x2000, setup burst DMA destination
address
3      bdf ERROR_ADDR_SETUP
4      ldi r0,0x64                     //loop counter is 100
5
//MAIN LOOP TRANSFER
6      loop 3,0
```

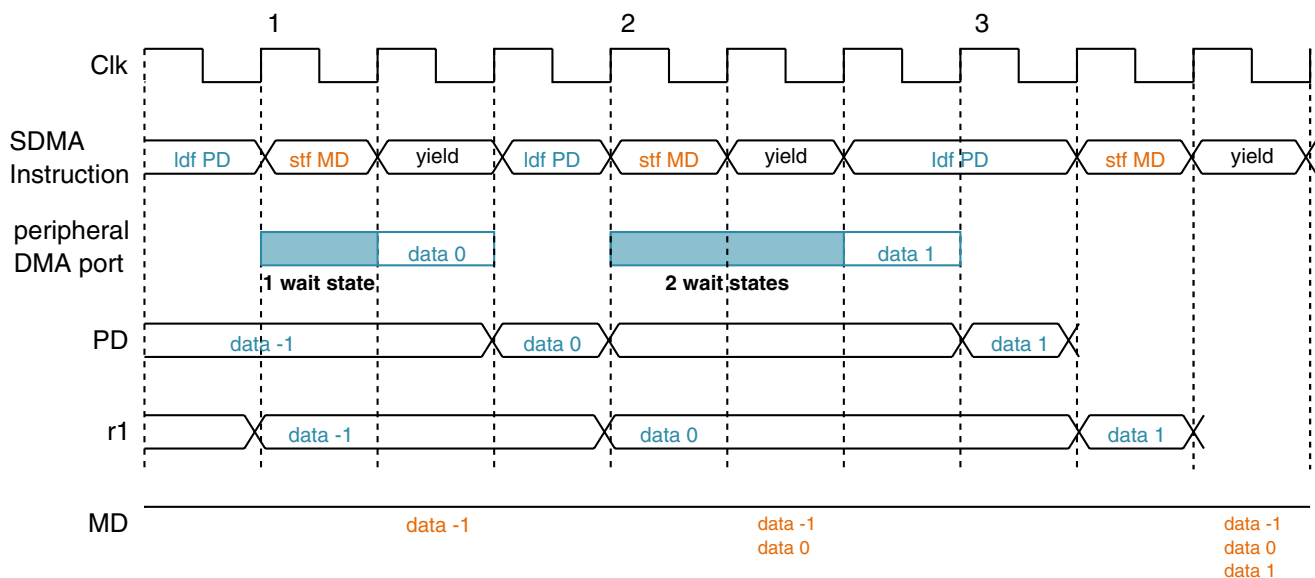
## Application Notes

```

7          ldf r1,PD|PF          // read 32 bits of PD and initiate a new read
access.
8          stf r1,MD|32          // store 32 bits of r1 in the MD fifo.
9          yield
10         ldf r1,PD             // last word data is read
11         stf r1,MD|32|FL       // to flush all remaining bytes of MD

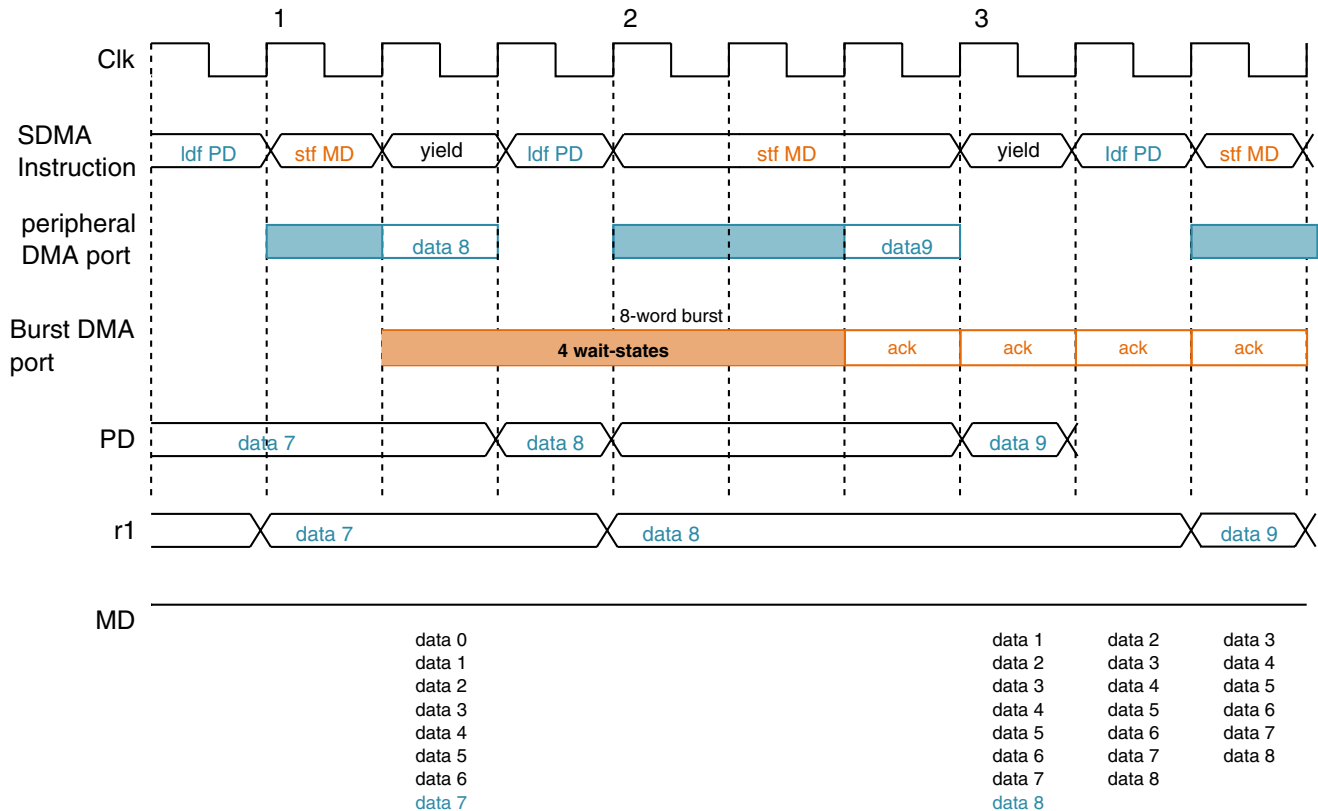
```

On instruction 1, the source address register of the peripheral DMA is programmed and data is fetched. This data is stored in PD and the SDMA reads PD during instruction 7, which is a one-cycle instruction that is read-access finished. On the same instruction (7), a data prefetch is required and a read access to the source peripheral is executed. In parallel, the SDMA stored the previous read data into the data register of MD. When MD (which is an eight-word FIFO) is full, a burst write access is executed to empty the FIFO. As long as the next SDMA instructions do not access the burst DMA, they will be one-cycle instructions. The following figures show how the peripheral DMA and burst DMA work in parallel.



**Figure 41-21. Peripheral to External Memory Example (1)**

As seen in the figure above, the read access triggered by the `ldf PD` instruction is symbolized by the blue bar when in progress. After wait states, the read data (data 0, data 1) is stored in PD on the clk rising edge. On edge 2, data 0 is available in PD so it can be transferred to the SDMA general register r1, and then stored in MD FIFO. On edge 3, data 1 is not in PD; therefore, SDMA is stalled on the `ldf` instruction, which lasts two cycles. The figure below shows an example of when MD FIFO is full with data.



**Figure 41-22. Peripheral to External Memory Example (2)**

In the previous figure, the write bar means the burst DMA is performing a write burst access. The latency to have the first write acknowledge is four cycles. SDMA is stalled on instruction `stf` because no acknowledge was received, MD FIFO is full, and there is no empty slot to store `data 9`. When an acknowledge is sampled by the burst DMA, FIFO is shifted and `data 8` is written. As long as there is at least one empty slot in MD FIFO, the `stf MD` instruction lasts one cycle.

#### 41.7.2.3.2 External Memory to Peripheral Transfer

A transfer from the external memory to a peripheral involves the peripheral DMA and the burst DMA.

The code for transferring 100 word from external memory to a word peripheral would be as follows:

##### External Memory to Peripheral Transfer

```
//SETUP SECTION source and destination addresses are already in r1 and r2
1      stf r1, MSA|PF           //r1=0x1000 and starts a 8-word read burst
2      stf r2, PDA|SZ32|P       //r2=0x2010, setup peripheral DMA destination address
3      bdf ERROR_ADDR_SETUP
4      ldi r0,0x64              //loop counter is 100
//MAIN LOOP TRANSFER
6      loop 3,0
```

```

7      ldf r1,MD|32|PF          // read 32 bits of MD and initiate a new read access
                                   // if MD is empty after this reading.
8      stf r1,PD                // store 32 bits of r1 in the PD.
9      yield
10     ldf r1,MD|32             // last word data is read
11     stf r1,PD                // last write access

```

On instruction 1, a read burst of 8 words begins. Read data is staged into MD. On instruction 7 (and if data is available in MD), 32 bits are copied into r1. Then instruction 8 writes them into PD and an automatic flush is executed. The SDMA core, peripheral DMA, and burst DMA can work in parallel as long as no SDMA instruction tries to start a new write access on the peripheral DMA while the previous access is still in progress, or as long as there is data in MD when the SDMA tries to read it.

#### 41.7.2.4 Transfer Between External Memory and Internal Memory

Since the internal memory (ARM platform RAM) is accessed via the peripheral DMA and the external memory is accessed via the burst DMA, the SDMA scripts that are described in [Transfer Between Peripheral and External Memory](#) can be reused. The exception is that the peripheral DMA address registers (PSA or PDA, depending on the script) should be programmed in incremented mode rather than frozen mode.

##### 41.7.2.4.1 Internal Memory to Internal Memory

The internal memory can only be accessed via the peripheral DMA, so the script described in [Peripheral to Peripheral Transfer](#) can be reused with a different programming of the peripheral DMA address registers.

##### 41.7.2.4.2 Transfer Between Peripheral and Internal Memory

For this transfer, the peripheral DMA is also used in copy mode.

The SDMA script is very similar to the one described in [Peripheral to Peripheral Transfer](#), except for the peripheral DMA address registers programming.

## 41.8 ARM Platform Memory Map and Control Register Definitions

The ARM platform controls the SDMA by means of several interface registers. Those registers are described in the current section.

All registers are clocked with the SDMA clock (which means the ARM platform must ensure that the SDMA clock is running when it wants to access any register).

**SDMAARM memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20E_C000	ARM platform Channel 0 Pointer (SDMAARM_MC0PTR)	32	R/W	0000_0000h	<a href="#">41.8.1/2606</a>
20E_C004	Channel Interrupts (SDMAARM_INTR)	32	w1c	0000_0000h	<a href="#">41.8.2/2606</a>
20E_C008	Channel Stop/Channel Status (SDMAARM_STOP_STAT)	32	w1c	0000_0000h	<a href="#">41.8.3/2606</a>
20E_C00C	Channel Start (SDMAARM_HSTART)	32	R/W	0000_0000h	<a href="#">41.8.4/2607</a>
20E_C010	Channel Event Override (SDMAARM_EVTOVR)	32	R/W	0000_0000h	<a href="#">41.8.5/2607</a>
20E_C014	Channel BP Override (SDMAARM_DSPOVR)	32	R/W	FFFF_FFFFh	<a href="#">41.8.6/2608</a>
20E_C018	Channel ARM platform Override (SDMAARM_HOSTOVR)	32	R/W	0000_0000h	<a href="#">41.8.7/2608</a>
20E_C01C	Channel Event Pending (SDMAARM_EVTPEND)	32	w1c	0000_0000h	<a href="#">41.8.8/2608</a>
20E_C024	Reset Register (SDMAARM_RESET)	32	R	0000_0000h	<a href="#">41.8.9/2609</a>
20E_C028	DMA Request Error Register (SDMAARM_EVTERR)	32	R	0000_0000h	<a href="#">41.8.10/2610</a>
20E_C02C	Channel ARM platform Interrupt Mask (SDMAARM_INTRMASK)	32	R/W	0000_0000h	<a href="#">41.8.11/2610</a>
20E_C030	Schedule Status (SDMAARM_PSW)	32	R	0000_0000h	<a href="#">41.8.12/2611</a>
20E_C034	DMA Request Error Register (SDMAARM_EVTERRDBG)	32	R	0000_0000h	<a href="#">41.8.13/2611</a>
20E_C038	Configuration Register (SDMAARM_CONFIG)	32	R/W	0000_0003h	<a href="#">41.8.14/2612</a>
20E_C03C	SDMA LOCK (SDMAARM_SDMA_LOCK)	32	R/W	0000_0000h	<a href="#">41.8.15/2613</a>
20E_C040	OnCE Enable (SDMAARM_ONCE_ENB)	32	R/W	0000_0000h	<a href="#">41.8.16/2614</a>
20E_C044	OnCE Data Register (SDMAARM_ONCE_DATA)	32	R/W	0000_0000h	<a href="#">41.8.17/2615</a>
20E_C048	OnCE Instruction Register (SDMAARM_ONCE_INSTR)	32	R/W	0000_0000h	<a href="#">41.8.18/2615</a>
20E_C04C	OnCE Status Register (SDMAARM_ONCE_STAT)	32	R	0000_E000h	<a href="#">41.8.19/2615</a>
20E_C050	OnCE Command Register (SDMAARM_ONCE_CMD)	32	R/W	0000_0000h	<a href="#">41.8.20/2617</a>

*Table continues on the next page...*

**SDMAARM memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/ page</b>
20E_C058	Illegal Instruction Trap Address (SDMAARM_ILLINSTADDR)	32	R/W	0000_0001h	<a href="#">41.8.21/ 2618</a>
20E_C05C	Channel 0 Boot Address (SDMAARM_CHN0ADDR)	32	R/W	0000_0050h	<a href="#">41.8.22/ 2618</a>
20E_C060	DMA Requests (SDMAARM_EVT_MIRROR)	32	R	0000_0000h	<a href="#">41.8.23/ 2619</a>
20E_C064	DMA Requests 2 (SDMAARM_EVT_MIRROR2)	32	R	0000_0000h	<a href="#">41.8.24/ 2619</a>
20E_C070	Cross-Trigger Events Configuration Register 1 (SDMAARM_XTRIG_CONF1)	32	R/W	0000_0000h	<a href="#">41.8.25/ 2620</a>
20E_C074	Cross-Trigger Events Configuration Register 2 (SDMAARM_XTRIG_CONF2)	32	R/W	0000_0000h	<a href="#">41.8.26/ 2622</a>
20E_C100	Channel Priority Registers (SDMAARM_SDMA_CHNPRI0)	32	R/W	0000_0000h	<a href="#">41.8.27/ 2623</a>
20E_C104	Channel Priority Registers (SDMAARM_SDMA_CHNPRI1)	32	R/W	0000_0000h	<a href="#">41.8.27/ 2623</a>
20E_C108	Channel Priority Registers (SDMAARM_SDMA_CHNPRI2)	32	R/W	0000_0000h	<a href="#">41.8.27/ 2623</a>
20E_C10C	Channel Priority Registers (SDMAARM_SDMA_CHNPRI3)	32	R/W	0000_0000h	<a href="#">41.8.27/ 2623</a>
20E_C110	Channel Priority Registers (SDMAARM_SDMA_CHNPRI4)	32	R/W	0000_0000h	<a href="#">41.8.27/ 2623</a>
20E_C114	Channel Priority Registers (SDMAARM_SDMA_CHNPRI5)	32	R/W	0000_0000h	<a href="#">41.8.27/ 2623</a>
20E_C118	Channel Priority Registers (SDMAARM_SDMA_CHNPRI6)	32	R/W	0000_0000h	<a href="#">41.8.27/ 2623</a>
20E_C11C	Channel Priority Registers (SDMAARM_SDMA_CHNPRI7)	32	R/W	0000_0000h	<a href="#">41.8.27/ 2623</a>
20E_C120	Channel Priority Registers (SDMAARM_SDMA_CHNPRI8)	32	R/W	0000_0000h	<a href="#">41.8.27/ 2623</a>
20E_C124	Channel Priority Registers (SDMAARM_SDMA_CHNPRI9)	32	R/W	0000_0000h	<a href="#">41.8.27/ 2623</a>
20E_C128	Channel Priority Registers (SDMAARM_SDMA_CHNPRI10)	32	R/W	0000_0000h	<a href="#">41.8.27/ 2623</a>
20E_C12C	Channel Priority Registers (SDMAARM_SDMA_CHNPRI11)	32	R/W	0000_0000h	<a href="#">41.8.27/ 2623</a>
20E_C130	Channel Priority Registers (SDMAARM_SDMA_CHNPRI12)	32	R/W	0000_0000h	<a href="#">41.8.27/ 2623</a>
20E_C134	Channel Priority Registers (SDMAARM_SDMA_CHNPRI13)	32	R/W	0000_0000h	<a href="#">41.8.27/ 2623</a>
20E_C138	Channel Priority Registers (SDMAARM_SDMA_CHNPRI14)	32	R/W	0000_0000h	<a href="#">41.8.27/ 2623</a>
20E_C13C	Channel Priority Registers (SDMAARM_SDMA_CHNPRI15)	32	R/W	0000_0000h	<a href="#">41.8.27/ 2623</a>

*Table continues on the next page...*

**SDMAARM memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20E_C140	Channel Priority Registers (SDMAARM_SDMA_CHNPRI16)	32	R/W	0000_0000h	<a href="#">41.8.27/2623</a>
20E_C144	Channel Priority Registers (SDMAARM_SDMA_CHNPRI17)	32	R/W	0000_0000h	<a href="#">41.8.27/2623</a>
20E_C148	Channel Priority Registers (SDMAARM_SDMA_CHNPRI18)	32	R/W	0000_0000h	<a href="#">41.8.27/2623</a>
20E_C14C	Channel Priority Registers (SDMAARM_SDMA_CHNPRI19)	32	R/W	0000_0000h	<a href="#">41.8.27/2623</a>
20E_C150	Channel Priority Registers (SDMAARM_SDMA_CHNPRI20)	32	R/W	0000_0000h	<a href="#">41.8.27/2623</a>
20E_C154	Channel Priority Registers (SDMAARM_SDMA_CHNPRI21)	32	R/W	0000_0000h	<a href="#">41.8.27/2623</a>
20E_C158	Channel Priority Registers (SDMAARM_SDMA_CHNPRI22)	32	R/W	0000_0000h	<a href="#">41.8.27/2623</a>
20E_C15C	Channel Priority Registers (SDMAARM_SDMA_CHNPRI23)	32	R/W	0000_0000h	<a href="#">41.8.27/2623</a>
20E_C160	Channel Priority Registers (SDMAARM_SDMA_CHNPRI24)	32	R/W	0000_0000h	<a href="#">41.8.27/2623</a>
20E_C164	Channel Priority Registers (SDMAARM_SDMA_CHNPRI25)	32	R/W	0000_0000h	<a href="#">41.8.27/2623</a>
20E_C168	Channel Priority Registers (SDMAARM_SDMA_CHNPRI26)	32	R/W	0000_0000h	<a href="#">41.8.27/2623</a>
20E_C16C	Channel Priority Registers (SDMAARM_SDMA_CHNPRI27)	32	R/W	0000_0000h	<a href="#">41.8.27/2623</a>
20E_C170	Channel Priority Registers (SDMAARM_SDMA_CHNPRI28)	32	R/W	0000_0000h	<a href="#">41.8.27/2623</a>
20E_C174	Channel Priority Registers (SDMAARM_SDMA_CHNPRI29)	32	R/W	0000_0000h	<a href="#">41.8.27/2623</a>
20E_C178	Channel Priority Registers (SDMAARM_SDMA_CHNPRI30)	32	R/W	0000_0000h	<a href="#">41.8.27/2623</a>
20E_C17C	Channel Priority Registers (SDMAARM_SDMA_CHNPRI31)	32	R/W	0000_0000h	<a href="#">41.8.27/2623</a>
20E_C200	Channel Enable RAM (SDMAARM_CHNENBL0)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C204	Channel Enable RAM (SDMAARM_CHNENBL1)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C208	Channel Enable RAM (SDMAARM_CHNENBL2)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C20C	Channel Enable RAM (SDMAARM_CHNENBL3)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C210	Channel Enable RAM (SDMAARM_CHNENBL4)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C214	Channel Enable RAM (SDMAARM_CHNENBL5)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>

*Table continues on the next page...*

**SDMAARM memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20E_C218	Channel Enable RAM (SDMAARM_CHNENBL6)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C21C	Channel Enable RAM (SDMAARM_CHNENBL7)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C220	Channel Enable RAM (SDMAARM_CHNENBL8)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C224	Channel Enable RAM (SDMAARM_CHNENBL9)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C228	Channel Enable RAM (SDMAARM_CHNENBL10)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C22C	Channel Enable RAM (SDMAARM_CHNENBL11)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C230	Channel Enable RAM (SDMAARM_CHNENBL12)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C234	Channel Enable RAM (SDMAARM_CHNENBL13)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C238	Channel Enable RAM (SDMAARM_CHNENBL14)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C23C	Channel Enable RAM (SDMAARM_CHNENBL15)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C240	Channel Enable RAM (SDMAARM_CHNENBL16)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C244	Channel Enable RAM (SDMAARM_CHNENBL17)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C248	Channel Enable RAM (SDMAARM_CHNENBL18)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C24C	Channel Enable RAM (SDMAARM_CHNENBL19)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C250	Channel Enable RAM (SDMAARM_CHNENBL20)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C254	Channel Enable RAM (SDMAARM_CHNENBL21)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C258	Channel Enable RAM (SDMAARM_CHNENBL22)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C25C	Channel Enable RAM (SDMAARM_CHNENBL23)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C260	Channel Enable RAM (SDMAARM_CHNENBL24)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C264	Channel Enable RAM (SDMAARM_CHNENBL25)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C268	Channel Enable RAM (SDMAARM_CHNENBL26)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C26C	Channel Enable RAM (SDMAARM_CHNENBL27)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>

*Table continues on the next page...*



**SDMAARM memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/page</b>
20E_C270	Channel Enable RAM (SDMAARM_CHNENBL28)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C274	Channel Enable RAM (SDMAARM_CHNENBL29)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C278	Channel Enable RAM (SDMAARM_CHNENBL30)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C27C	Channel Enable RAM (SDMAARM_CHNENBL31)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C280	Channel Enable RAM (SDMAARM_CHNENBL32)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C284	Channel Enable RAM (SDMAARM_CHNENBL33)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C288	Channel Enable RAM (SDMAARM_CHNENBL34)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C28C	Channel Enable RAM (SDMAARM_CHNENBL35)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C290	Channel Enable RAM (SDMAARM_CHNENBL36)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C294	Channel Enable RAM (SDMAARM_CHNENBL37)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C298	Channel Enable RAM (SDMAARM_CHNENBL38)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C29C	Channel Enable RAM (SDMAARM_CHNENBL39)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C2A0	Channel Enable RAM (SDMAARM_CHNENBL40)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C2A4	Channel Enable RAM (SDMAARM_CHNENBL41)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C2A8	Channel Enable RAM (SDMAARM_CHNENBL42)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C2AC	Channel Enable RAM (SDMAARM_CHNENBL43)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C2B0	Channel Enable RAM (SDMAARM_CHNENBL44)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C2B4	Channel Enable RAM (SDMAARM_CHNENBL45)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C2B8	Channel Enable RAM (SDMAARM_CHNENBL46)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>
20E_C2BC	Channel Enable RAM (SDMAARM_CHNENBL47)	32	R/W	0000_0000h	<a href="#">41.8.28/2623</a>

## 41.8.1 ARM platform Channel 0 Pointer (SDMAARM\_MC0PTR)

Address: 20E\_C000h base + 0h offset = 20E\_C000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MC0PTR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SDMAARM\_MC0PTR field descriptions

Field	Description
31–0 MC0PTR	<b>Channel 0 Pointer</b> contains the 32-bit address, in ARM platform memory, of channel 0 control block (the boot channel). Appendix A fully describes the SDMA Application Programming Interface (API). The ARM platform has a read/write access and the SDMA has a read-only access.

## 41.8.2 Channel Interrupts (SDMAARM\_INTR)

Address: 20E\_C000h base + 4h offset = 20E\_C004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HI[31:0]																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SDMAARM\_INTR field descriptions

Field	Description
31–0 HI[31:0]	The ARM platform Interrupts register contains the 32 HI[i] bits. If any bit is set, it will cause an interrupt to the ARM platform. This register is a "write-ones" register to the ARM platform. When the ARM platform sets a bit in this register the corresponding HI[i] bit is cleared. The interrupt service routine should clear individual channel bits when their interrupts are serviced, failure to do so will cause continuous interrupts. The SDMA is responsible for setting the HI[i] bit corresponding to the current channel when the corresponding <code>done</code> instruction is executed.

## 41.8.3 Channel Stop/Channel Status (SDMAARM\_STOP\_STAT)

Address: 20E\_C000h base + 8h offset = 20E\_C008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HE																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SDMAARM\_STOP\_STAT field descriptions**

Field	Description
31–0 HE	This 32-bit register gives access to the ARM platform Enable bits. There is one bit for every channel. This register is a "write-ones" register to the ARM platform. When the ARM platform writes 1 in bit <i>i</i> of this register, it clears the HE[ <i>i</i> ] and HSTART[ <i>i</i> ] bits. Reading this register yields the current state of the HE[ <i>i</i> ] bits.

**41.8.4 Channel Start (SDMAARM\_HSTART)**

Address: 20E\_C000h base + Ch offset = 20E\_C00Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HSTART_HE																															
W	w1c																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SDMAARM\_HSTART field descriptions**

Field	Description
31–0 HSTART_HE	<p>The HSTART_HE registers are 32 bits wide with one bit for every channel. When a bit is written to 1, it enables the corresponding channel. Two physical registers are accessed with that address (HSTART and HE), which enables the ARM platform to trigger a channel a second time before the first trigger is processed.</p> <ul style="list-style-type: none"> <li>This register is a "write-ones" register to the ARM platform. Neither HSTART[<i>i</i>] bit can be set while the corresponding HE[<i>i</i>] bit is cleared.</li> <li>When the ARM platform tries to set the HSTART[<i>i</i>] bit by writing a one (if the corresponding HE[<i>i</i>] bit is clear), the bit in the HSTART[<i>i</i>] register will remain cleared and the HE[<i>i</i>] bit will be set.</li> <li>If the corresponding HE[<i>i</i>] bit was already set, the HSTART[<i>i</i>] bit will be set. The next time the SDMA channel <i>i</i> attempts to clear the HE[<i>i</i>] bit by means of a <code>done</code> instruction, the bit in the HSTART[<i>i</i>] register will be cleared and the HE[<i>i</i>] bit will take the old value of the HSTART[<i>i</i>] bit.</li> <li>Reading this register yields the current state of the HSTART[<i>i</i>] bits. This mechanism enables the ARM platform to pipeline two HSTART commands per channel.</li> </ul>

**41.8.5 Channel Event Override (SDMAARM\_EVTOVR)**

Address: 20E\_C000h base + 10h offset = 20E\_C010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EO																															
W	EO																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SDMAARM\_EVTOVR field descriptions**

Field	Description
31–0 EO	The Channel Event Override register contains the 32 EO[ <i>i</i> ] bits. A bit set in this register causes the SDMA to ignore DMA requests when scheduling the corresponding channel.

## 41.8.6 Channel BP Override (SDMAARM\_DSPOVR)

Address: 20E\_C000h base + 14h offset = 20E\_C014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W	DO																															
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

### SDMAARM\_DSPOVR field descriptions

Field	Description
31–0 DO	This register is reserved. All DO bits should be set to the reset value of 1. A setting of 0 will prevent SDMA channels from starting according to the condition described in <a href="#">Runnable Channels Evaluation</a> .  0 - Reserved 1 - Reset value.

## 41.8.7 Channel ARM platform Override (SDMAARM\_HOSTOVR)

Address: 20E\_C000h base + 18h offset = 20E\_C018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W	HO																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SDMAARM\_HOSTOVR field descriptions

Field	Description
31–0 HO	The Channel ARM platform Override register contains the 32 HO[i] bits. A bit set in this register causes the SDMA to ignore the ARM platform enable bit (HE) when scheduling the corresponding channel.

## 41.8.8 Channel Event Pending (SDMAARM\_EVTPEND)

Address: 20E\_C000h base + 1Ch offset = 20E\_C01Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EP																															
W	w1c																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SDMAARM\_EVTPEND field descriptions**

Field	Description
31–0 EP	<p>The Channel Event Pending register contains the 32 EP[i] bits. Reading this register enables the ARM platform to determine what channels are pending after the reception of a DMA request.</p> <ul style="list-style-type: none"> <li>Setting a bit in this register causes the SDMA to reevaluate scheduling as if a DMA request mapped on this channel had occurred. This is useful for starting up channels, so that initialization is done before awaiting the first request. The scheduler can also set bits in the EVTPEND register according to the received DMA requests.</li> <li>The EP[i] bit may be cleared by the <code>done</code> instruction when running the channel <i>i</i> script. This is a "write-ones" mechanism: Writing a '0' does not clear the corresponding bit.</li> </ul>

**41.8.9 Reset Register (SDMAARM\_RESET)**

Address: 20E\_C000h base + 24h offset = 20E\_C024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												RESCHED		RESET	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SDMAARM\_RESET field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.

*Table continues on the next page...*

**SDMAARM\_RESET field descriptions (continued)**

Field	Description
1 RESCHED	When set, this bit forces the SDMA to reschedule as if a script had executed a <code>done</code> instruction. This enables the ARM platform to recover from a runaway script on a channel by clearing its HE[i] bit via the STOP register, and then forcing a reschedule via the RESCHED bit. The RESCHED bit is cleared when the context switch starts.
0 RESET	When set, this bit causes the SDMA to be held in a software reset. The internal reset signal is held low 16 cycles; the RESET bit is automatically cleared when the internal reset signal rises.

**41.8.10 DMA Request Error Register (SDMAARM\_EVTERR)**

Address: 20E\_C000h base + 28h offset = 20E\_C028h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CHNERR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SDMAARM\_EVTERR field descriptions**

Field	Description
31–0 CHNERR	<p>This register is used by the SDMA to warn the ARM platform when an incoming DMA request was detected and it triggers a channel that is already pending or being serviced. This probably means there is an overflow of data for that channel.</p> <ul style="list-style-type: none"> <li>An interrupt is sent to the ARM platform if the corresponding channel bit is set in the INTRMASK register.</li> <li>This is a "write-ones" register for the scheduler. It is only able to set the flags. The flags are cleared when the register is read by the ARM platform or during SDMA reset.</li> <li>The CHNERR[i] bit is set when a DMA request that triggers channel <i>i</i> is received through the corresponding input pins and the EP[i] bit is already set; the EVTERR[i] bit is unaffected if the ARM platform tries to set the EP[i] bit, whereas, that EP[i] bit is already set.</li> </ul>

**41.8.11 Channel ARM platform Interrupt Mask (SDMAARM\_INTRMASK)**

Address: 20E\_C000h base + 2Ch offset = 20E\_C02Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	HIMASK																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SDMAARM\_INTRMASK field descriptions**

Field	Description
31–0 HIMASK	The Interrupt Mask Register contains 32 interrupt generation mask bits. If bit HIMASK[i] is set, the HI[i] bit is set and an interrupt is sent to the ARM platform when a DMA request error is detected on channel <i>i</i> (for example, EVTERR[i] is set).

**41.8.12 Schedule Status (SDMAARM\_PSW)**

Address: 20E\_C000h base + 30h offset = 20E\_C030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																NCP[2:0]			NCR[4:0]				CCP[2:0]			CCR[4:0]					
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SDMAARM\_PSW field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–13 NCP[2:0]	The Next Channel Priority gives the next pending channel priority. When the priority is 0, it means there is no pending channel and the NCR value has no meaning.  0 No running channel 1 Active channel priority
12–8 NCR[4:0]	The Next Channel Register indicates the number of the next scheduled pending channel with the highest priority.
7–4 CCP[2:0]	The Current Channel Priority indicates the priority of the current active channel. When the priority is 0, no channel is running: The SDMA is idle and the CCR value has no meaning. In the case that the SDMA has finished running the channel and has entered sleep state, CCP will indicate the priority of previous running channel.  0 No running channel 1 Active channel priority
3–0 CCR[4:0]	The Current Channel Register indicates the number of the channel that is being executed by the SDMA. SDMA. In the case that the SDMA has finished running the channel and has entered sleep state, CCR will indicate the previous running channel.

**41.8.13 DMA Request Error Register (SDMAARM\_EVTERRDBG)**

Address: 20E\_C000h base + 34h offset = 20E\_C034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CHNERR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SDMAARM\_EVTERDBG field descriptions**

Field	Description
31–0 CHNERR	This register is the same as EVTERR, except reading it does not clear its contents. This address is meant to be used in debug mode. The ARM platform OnCE may check this register value without modifying it.

**41.8.14 Configuration Register (SDMAARM\_CONFIG)**

Address: 20E\_C000h base + 38h offset = 20E\_C038h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	0			DSPDMA	RTDOBS	0							ACR	0		CSM	
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	

**SDMAARM\_CONFIG field descriptions**

Field	Description
31–13 Reserved	This read-only field is reserved and always has the value 0.
12 DSPDMA	This bit's function is reserved and should be configured as zero. 0 - Reset Value 1 - Reserved
11 RTDOBS	Indicates if Real-Time Debug pins are used: They do not toggle by default in order to reduce power consumption. 0 RTD pins disabled 1 RTD pins enabled
10–5 Reserved	This read-only field is reserved and always has the value 0.
4 ACR	ARM platform DMA / SDMA Core Clock Ratio. Selects the clock ratio between ARM platform DMA interfaces (burst DMA and peripheral DMA) and the internal SDMA core clock. The frequency selection is determined separately by the chip clock controller. This bit has to match the configuration of the chip clock controller that generates the clocks used in the SDMA. 0 ARM platform DMA interface frequency equals twice core frequency 1 ARM platform DMA interface frequency equals core frequency
3–2 Reserved	This read-only field is reserved and always has the value 0.

*Table continues on the next page...*



**SDMAARM\_CONFIG field descriptions (continued)**

Field	Description
1–0 CSM	<p>Selects the Context Switch Mode. The ARM platform has a read/write access. The SDMA cannot modify that register. The value at reset is 3, which selects the dynamic context switch by default. That register can be modified at anytime but the new context switch configuration will only be taken into account at the start of the next restore phase.</p> <p>NOTE: The first call to SDMA's channel 0 Bootload script after reset should use static context switch mode to ensure the context RAM for channel 0 is initialized in the channel SAVE Phase. After Channel 0 is run once, then any of the dynamic context modes can be used.</p> <p>0 static 1 dynamic low power 2 dynamic with no loop 3 dynamic</p>

**41.8.15 SDMA LOCK (SDMAARM\_SDMA\_LOCK)**

Address: 20E\_C000h base + 3Ch offset = 20E\_C03Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0														SRESET_LOCK_ CLR	LOCK
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SDMAARM\_SDMA\_LOCK field descriptions**

Field	Description
31–2 Reserved	This read-only field is reserved and always has the value 0.
1 SRESET_LOCK_CLR	<p>The SRESET_LOCK_CLR bit determine if the LOCK bit is cleared on a software reset triggered by writing to the RESET register. This bit cannot be changed if LOCK=1. SRESET_LOCK_CLR is cleared by conditions that clear the LOCK bit.</p> <p>0 Software Reset does not clear the LOCK bit. 1 Software Reset clears the LOCK bit.</p>

*Table continues on the next page...*

**SDMAARM\_SDMA\_LOCK field descriptions (continued)**

Field	Description
0 LOCK	<p>The LOCK bit is used to restrict access to update SDMA script memory through ROM channel zero scripts and through the OnCE interface under ARM platform control.</p> <p>The LOCK bit is set:</p> <ul style="list-style-type: none"> <li>• The SDMA_LOCK, ONCE_ENB, CH0ADDR, and ILLINSTADDR registers cannot be written. These registers can be read, but writes are ignored.</li> <li>• SDMA software executing out of ROM or RAM may check the LOCK bit in the LOCK register <a href="#">Lock Status Register (SDMACORE_SDMA_LOCK)</a> to determine if certain operations are allowed, such as up-loading new scripts.</li> </ul> <p>Once the LOCK bit is set to 1, only a reset can clear it. The LOCK bit is cleared by a hardware reset. LOCK is cleared by a software reset only if SRESET_LOCK_CLR is set.</p> <p>0 LOCK disengaged. 1 LOCK enabled.</p>

**41.8.16 OnCE Enable (SDMAARM\_ONCE\_ENB)**

Address: 20E\_C000h base + 40h offset = 20E\_C040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															ENB
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SDMAARM\_ONCE\_ENB field descriptions**

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 ENB	<p>The OnCE Enable register selects the OnCE control source: When cleared (0), the OnCE registers are accessed through the JTAG interface; when set (1), the OnCE registers may be accessed by the ARM platform through the addresses described, as follows.</p> <ul style="list-style-type: none"> <li>• After reset, the OnCE registers are accessed through the JTAG interface.</li> <li>• Writing a 1 to ENB enables the ARM platform to access the ONCE_* as any other SDMA control register.</li> <li>• When cleared (0), all the ONCE_xxx registers cannot be written.</li> </ul> <p>The value of ENB cannot be changed if the LOCK bit in the SDMA_LOCK register is set.</p>

### 41.8.17 OnCE Data Register (SDMAARM\_ONCE\_DATA)

Address: 20E\_C000h base + 44h offset = 20E\_C044h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DATA																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SDMAARM\_ONCE\_DATA field descriptions

Field	Description
31–0 DATA	Data register of the OnCE JTAG controller. Refer to <a href="#">OnCE and Real-Time Debug</a> for information on this register.

### 41.8.18 OnCE Instruction Register (SDMAARM\_ONCE\_INSTR)

Address: 20E\_C000h base + 48h offset = 20E\_C048h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																INSTR															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SDMAARM\_ONCE\_INSTR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–0 INSTR	Instruction register of the OnCE JTAG controller. Refer to <a href="#">OnCE and Real-Time Debug</a> for information on this register.

### 41.8.19 OnCE Status Register (SDMAARM\_ONCE\_STAT)

Address: 20E\_C000h base + 4Ch offset = 20E\_C04Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PST[3:0]				RCV	EDR	ODR	SWB	MST	0				ECDR		
W																
Reset	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0

**SDMAARM\_ONCE\_STAT field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–12 PST[3:0]	<p>The Processor Status bits reflect the state of the SDMA RISC engine. Its states are as follows:</p> <ul style="list-style-type: none"> <li>• The "Program" state is the usual instruction execution cycle.</li> <li>• The "Data" state is inserted when there are wait-states during a load or a store on the data bus (ld or st).</li> <li>• The "Change of Flow" state is the second cycle of any instruction that breaks the sequence of instructions (jumps and channel switching instructions).</li> <li>• The "Change of Flow in Loop" state is used when an error causes a hardware loop exit.</li> <li>• The "Debug" state means the SDMA is in debug mode.</li> <li>• The "Functional Unit" state is inserted when there are wait-states during a load or a store on the functional units bus (ldf or stf).</li> <li>• In "Sleep" modes, no script is running (this is the RISC engine idle state). The "after Reset" is slightly different because no context restoring phase will happen when a channel is triggered: The script located at address 0 will be executed (boot operation).</li> <li>• The "in Sleep" states are the same as above except they do not have any corresponding channel: They are used when entering debug mode after reset. The reason is that it is necessary to return to the "Sleep after Reset" state when leaving debug mode.</li> </ul> <p>0 Program 1 Data 2 Change of Flow 3 Change of Flow in Loop 4 Debug 5 Functional Unit 6 Sleep 7 Save 8 Program in Sleep 9 Data in Sleep 10 Change of Flow in Sleep 11 Change Flow in Loop in Sleep 12 Debug in Sleep 13 Functional Unit in Sleep 14 Sleep after Reset 15 Restore</p>
11 RCV	After each write access to the real time buffer (RTB), the RCV bit is set. This bit is cleared after execution of an <code>rbuffer</code> command and on a JTAG reset.
10 EDR	This flag is raised when the SDMA has entered debug mode after an external debug request.
9 ODR	This flag is raised when the SDMA has entered debug mode after a OnCE debug request.
8 SWB	This flag is raised when the SDMA has entered debug mode after a software breakpoint.
7 MST	<p>This flag is raised when the OnCE is controlled from the ARM platform peripheral interface.</p> <p>0 The JTAG interface controls the OnCE. 1 The ARM platform peripheral interface controls the OnCE.</p>
6–3 Reserved	This read-only field is reserved and always has the value 0.

*Table continues on the next page...*

**SDMAARM\_ONCE\_STAT field descriptions (continued)**

Field	Description
2–0 ECDR	<p>Event Cell Debug Request. If the debug request comes from the event cell, the reason for entering debug mode is given by the EDR bits. If all three bits of the EDR are reset, then it did not generate any debug request. If the cell did generate a debug request, then at least one of the EDR bits is set (the meaning of the encoding is given below). The encoding of the EDR bits is useful to find out more precisely why the debug request was generated. A debug request from an event cell is generated for a specific combination of the addra_cond, addrb_cond, and data_cond conditions. The value of those fields is given by the EDR bits.</p> <p>0 1 matched addra_cond  1 1 matched addrb_cond  2 1 matched data_cond</p>

**41.8.20 OnCE Command Register (SDMAARM\_ONCE\_CMD)**

Address: 20E\_C000h base + 50h offset = 20E\_C050h

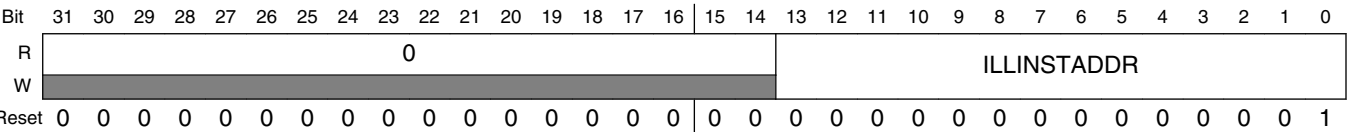
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																CMD															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SDMAARM\_ONCE\_CMD field descriptions**

Field	Description
31–4 Reserved	This read-only field is reserved and always has the value 0.
3–0 CMD	<p>Writing to this register will cause the OnCE to execute the command that is written. When needed, the ONCE_DATA and ONCE_INSTR registers should be loaded with the correct value before writing the command to that register. For a list of the OnCE commands and their usage, see <a href="#">OnCE and Real-Time Debug</a>.</p> <p><b>NOTE:</b> 7-15 reserved</p> <p>0 rstatus  1 dmov  2 exec_once  3 run_core  4 exec_core  5 debug_rqst  6 rbuffer</p>

### 41.8.21 Illegal Instruction Trap Address (SDMAARM\_ILLINSTADDR)

Address: 20E\_C000h base + 58h offset = 20E\_C058h

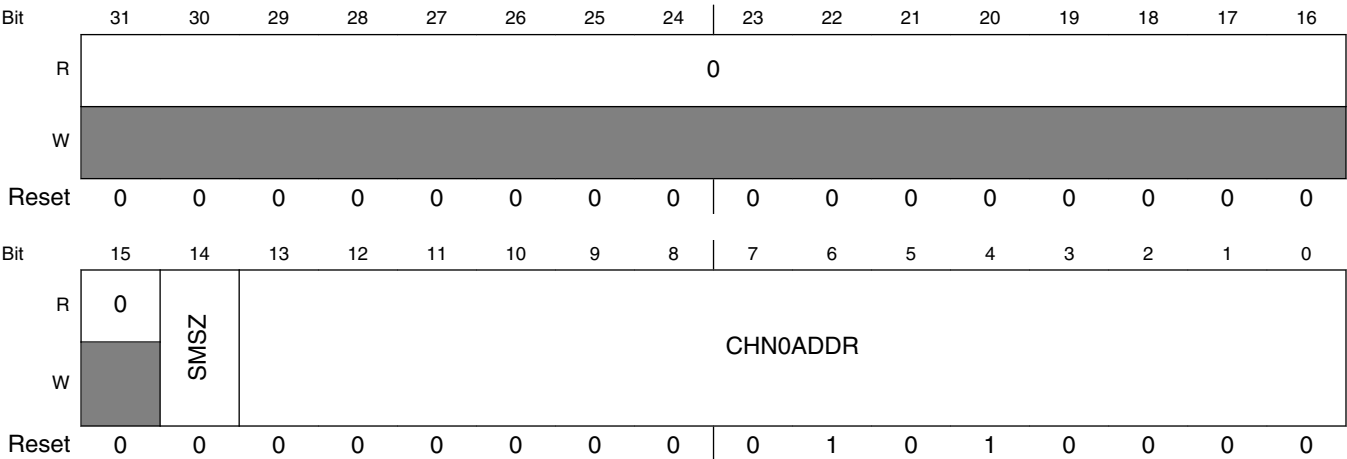


SDMAARM\_ILLINSTADDR field descriptions

Field	Description
31–14 Reserved	This read-only field is reserved and always has the value 0.
13–0 ILLINSTADDR	The Illegal Instruction Trap Address is the address where the SDMA jumps when an illegal instruction is executed. It is 0x0001 after reset.  The value of ILLINSTADDR cannot be changed if the LOCK bit in the SDMA_LOCK register is set.

### 41.8.22 Channel 0 Boot Address (SDMAARM\_CHN0ADDR)

Address: 20E\_C000h base + 5Ch offset = 20E\_C05Ch



SDMAARM\_CHN0ADDR field descriptions

Field	Description
31–15 Reserved	This read-only field is reserved and always has the value 0.
14 SMSZ	The bit 14 (Scratch Memory Size) determines if scratch memory must be available after every channel context. After reset, it is equal to 0, which defines a RAM space of 24 words for each channel. All of this area stores the channel context. By setting this bit, 32 words are reserved for every channel context,

Table continues on the next page...

**SDMAARM\_CHN0ADDR field descriptions (continued)**

Field	Description
	<p>which gives eight additional words that can be used by the channel script to store any type of data. Those words are never erased by the context switching mechanism.</p> <p>The value of SMSZ cannot be changed if the LOCK bit in the SDMA_LOCK register is set.</p> <p>0 24 words per context 1 32 words per context</p>
13–0 CHN0ADDR	<p>This 14-bit register is used by the boot code of the SDMA. After reset, it points to the standard boot routine in ROM (channel 0 routine). By changing this address, you can perform a boot sequence with your own routine. The very first instructions of the boot code fetch the contents of this register (it is also mapped in the SDMA memory space) and jump to the given address. The reset value is 0x0050 (decimal 80).</p> <p>The value of CHN0ADDR cannot be changed if the LOCK bit in the SDMA_LOCK register is set.</p>

**41.8.23 DMA Requests (SDMAARM\_EVT\_MIRROR)**

Address: 20E\_C000h base + 60h offset = 20E\_C060h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EVENTS																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SDMAARM\_EVT\_MIRROR field descriptions**

Field	Description
31–0 EVENTS	<p>This register reflects the DMA requests received by the SDMA for events 31-0. The ARM platform and the SDMA have a read-only access. There is one bit associated with each of 32 DMA request events. This information may be useful during debug of the blocks that generate the DMA requests. The EVT_MIRROR register is cleared following read access.</p> <p>0 DMA request event not pending 1 DMA request event pending</p>

**41.8.24 DMA Requests 2 (SDMAARM\_EVT\_MIRROR2)**

Address: 20E\_C000h base + 64h offset = 20E\_C064h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																EVENTS[47:32]															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SDMAARM\_EVT\_MIRROR2 field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–0 EVENTS[47:32]	This register reflects the DMA requests received by the SDMA for events 47-32. The ARM platform and the SDMA have a read-only access. There is one bit associated with each of DMA request events. This information may be useful during debug of the blocks that generate the DMA requests. The EVT_MIRROR2 register is cleared following read access.  0 - DMA request event not pending 1- DMA request event pending

**41.8.25 Cross-Trigger Events Configuration Register 1 (SDMAARM\_XTRIG\_CONF1)**

Address: 20E\_C000h base + 70h offset = 20E\_C070h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	CNF3	NUM3[5:0]						0	CNF2	NUM2[5:0]					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	CNF1	NUM1[5:0]						0	CNF0	NUM0[5:0]					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SDMAARM\_XTRIG\_CONF1 field descriptions**

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30 CNF3	Configuration of the SDMA event line number <i>i</i> that is connected to the cross-trigger. It determines whether the event line pulse is generated by the reception of a DMA request or by the starting of a channel script execution.  0 channel 1 DMA request
29–24 NUM3[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number <i>i</i> .
23 Reserved	This read-only field is reserved and always has the value 0.

*Table continues on the next page...*



**SDMAARM\_XTRIG\_CONF1 field descriptions (continued)**

Field	Description
22 CNF2	Configuration of the SDMA event line number <i>i</i> that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution.  0 channel 1 DMA request
21–16 NUM2[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number <i>i</i> .
15 Reserved	This read-only field is reserved and always has the value 0.
14 CNF1	Configuration of the SDMA event line number <i>i</i> that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution.  0 channel 1 DMA request
13–8 NUM1[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number <i>i</i> .
7 Reserved	This read-only field is reserved and always has the value 0.
6 CNF0	Configuration of the SDMA event line number <i>i</i> that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution.  0 channel 1 DMA request
5–0 NUM0[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number <i>i</i> .

## 41.8.26 Cross-Trigger Events Configuration Register 2 (SDMAARM\_XTRIG\_CONF2)

Address: 20E\_C000h base + 74h offset = 20E\_C074h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	CNF7	NUM7[5:0]							0	CNF6	NUM6[5:0]				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	CNF5	NUM5[5:0]							0	CNF4	NUM4[5:0]				
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SDMAARM\_XTRIG\_CONF2 field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30 CNF7	Configuration of the SDMA event line number <i>i</i> that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution.  0 channel 1 DMA request
29–24 NUM7[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number <i>i</i> .
23 Reserved	This read-only field is reserved and always has the value 0.
22 CNF6	Configuration of the SDMA event line number <i>i</i> that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution.  0 channel 1 DMA request
21–16 NUM6[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number <i>i</i> .
15 Reserved	This read-only field is reserved and always has the value 0.
14 CNF5	Configuration of the SDMA event line number <i>i</i> that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution

Table continues on the next page...

**SDMAARM\_XTRIG\_CONF2 field descriptions (continued)**

Field	Description
	0 channel 1 DMA request
13–8 NUM5[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number <i>i</i> .
7 Reserved	This read-only field is reserved and always has the value 0.
6 CNF4	Configuration of the SDMA event line number <i>i</i> that is connected to the cross-trigger. It determines whether the event line pulse is generated by receiving a DMA request or by starting a channel script execution.  0 channel 1 DMA request
5–0 NUM4[5:0]	Contains the number of the DMA request or channel that triggers the pulse on the cross-trigger event line number <i>i</i> .

**41.8.27 Channel Priority Registers (SDMAARM\_SDMA\_CHNPRIn)**

Address: 20E\_C000h base + 100h offset + (4d × *i*), where *i*=0d to 31d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																CHNPRIn															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SDMAARM\_SDMA\_CHNPRIn field descriptions**

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value 0.
2–0 CHNPRIn	This contains the priority of channel number <i>n</i> . Useful values are between 1 and 7; 0 is reserved by the SDMA hardware to determine when there is no pending channel. Reset value is 0, which prevents the channels from starting.

**41.8.28 Channel Enable RAM (SDMAARM\_CHNENBLn)**

Address: 20E\_C000h base + 200h offset + (4d × *i*), where *i*=0d to 47d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W	ENBLn																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

SDMAARM\_CHNENBL $n$  field descriptions

Field	Description
31–0 ENBL $n$	This 32-bit value selects the channels that are triggered by the DMA request number $n$ . If ENBL $n$ [ $i$ ] is set to 1, bit EP[ $i$ ] will be set when the DMA request $n$ is received. These 48 32-bit registers are physically located in a RAM, with no known reset value. It is thus essential for the ARM platform to program them before any DMA request is triggered to the SDMA, otherwise an unpredictable combination of channels may be started.

## 41.9 BP Memory Map and Control Register Definitions

The following section describes SDMA control registers available to the BP.

### NOTE

These registers are physically implemented in all platforms, but are not accessible when the SDMA BP control port is not connected. Reset values are calculated to allow the system to work when those registers cannot be accessed.

All registers are clocked with the SDMA clock (which means the SDMA clock must be running when the BP wants to access any register).

### SDMABP memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20E_C000	Channel 0 Pointer (SDMABP_DC0PTR)	32	R/W	0000_0000h	<a href="#">41.9.1/2624</a>
20E_C004	Channel Interrupts (SDMABP_INTR)	32	w1c	0000_0000h	<a href="#">41.9.2/2625</a>
20E_C008	Channel Stop/Channel Status (SDMABP_STOP_STAT)	32	R/W	0000_0000h	<a href="#">41.9.3/2625</a>
20E_C00C	Channel Start (SDMABP_DSTART)	32	R	0000_0000h	<a href="#">41.9.4/2626</a>
20E_C028	DMA Request Error Register (SDMABP_EVTERR)	32	R	0000_0000h	<a href="#">41.9.5/2626</a>
20E_C02C	Channel DSP Interrupt Mask (SDMABP_INTRMASK)	32	R/W	0000_0000h	<a href="#">41.9.6/2627</a>
20E_C034	DMA Request Error Register (SDMABP_EVTERRDBG)	32	R	0000_0000h	<a href="#">41.9.7/2627</a>

### 41.9.1 Channel 0 Pointer (SDMABP\_DC0PTR)

Address: 20E\_C000h base + 0h offset = 20E\_C000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SDMABP\_DC0PTR field descriptions**

Field	Description
31–0 DC0PTR	<b>Channel 0 Pointer</b> contains the 32-bit address, in BP memory, of the array of channel control blocks starting with the one for channel 0 (the control channel). This register should be initialized by the BP before it enables a channel (for example, channel 0). See the API document SDMA Scripts User Manual for the use of this register. The BP has a read/write access and the SDMA has a read-only access.

**41.9.2 Channel Interrupts (SDMABP\_INTR)**

Address: 20E\_C000h base + 4h offset = 20E\_C004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DI																															
W	w1c																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SDMABP\_INTR field descriptions**

Field	Description
31–0 DI	The BP Interrupts register contains the 32 DI[i] bits. If any bit is set, it will cause an interrupt to the BP. <ul style="list-style-type: none"> <li>This register is a "write-ones" register to the BP. When the BP sets a bit in this register, the corresponding DI[i] bit is cleared.</li> <li>The interrupt service routine should clear individual channel bits when their interrupts are serviced; failure to do so will cause continuous interrupts.</li> <li>The SDMA is responsible for setting the DI[i] bit corresponding to the current channel when the corresponding <code>done</code> instruction is executed.</li> </ul>

**41.9.3 Channel Stop/Channel Status (SDMABP\_STOP\_STAT)**

Address: 20E\_C000h base + 8h offset = 20E\_C008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																	DE															
W																	w1c															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SDMABP\_STOP\_STAT field descriptions**

Field	Description
31–0 DE	This 32-bit register gives access to the BP (DSP) Enable bits, DE. There is one bit for every channel. <ul style="list-style-type: none"> <li>This register is a "write-ones" register to the BP.</li> <li>When the BP writes 1 in bit <i>i</i> of this register, it clears the DE[i] and DSTART[i] bits.</li> <li>Reading this register yields the current state of the DE[i] bits.</li> </ul>

## 41.9.4 Channel Start (SDMABP\_DSTART)

Address: 20E\_C000h base + Ch offset = 20E\_C00Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DSTART_DE																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SDMABP\_DSTART field descriptions

Field	Description
31–0 DSTART_DE	<p>The DSTART_DE registers are 32 bits wide with one bit for every channel.</p> <ul style="list-style-type: none"> <li>When a bit is written to 1, it enables the corresponding channel.</li> <li>Two physical registers are accessed with that address (DSTART and DE), which enables the BP to trigger a channel a second time before the first trigger was processed.</li> <li>This register is a "write-ones" register to the BP. Neither DSTART[i] bit can be set while the corresponding DE[i] bit is cleared.</li> <li>When the BP tries to set the DSTART[i] bit by writing a one (if the corresponding DE[i] bit is clear), the bit in the DSTART[i] register will remain cleared and the DE[i] bit will be set. If the corresponding DE[i] bit was already set, the DSTART[i] bit will be set.</li> <li>The next time the SDMA channel <i>i</i> attempts to clear the DE[i] bit by means of a <code>done</code> instruction, the bit in the DSTART[i] register will be cleared and the DE[i] bit will take the old value of the DSTART[i] bit.</li> <li>Reading this register yields the current state of the DSTART[i] bits. This mechanism enables the BP to pipeline two DSTART commands per channel.</li> </ul>

## 41.9.5 DMA Request Error Register (SDMABP\_EVTERR)

Address: 20E\_C000h base + 28h offset = 20E\_C028h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CHNERR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SDMABP\_EVTERR field descriptions

Field	Description
31–0 CHNERR	<p>This register is used by the SDMA to warn the BP when an incoming DMA request was detected; it then triggers a channel that is already pending or being serviced, which may mean there is an overflow of data for that channel. An interrupt is sent to the BP if the corresponding channel bit is set in the INTRMASK register.</p> <ul style="list-style-type: none"> <li>This is a "write-ones" register for the scheduler. It is only able to set the flags. The flags are cleared when the register is read by the BP or during an SDMA reset.</li> <li>The CHNERR[i] bit is set when a DMA request that triggers channel <i>i</i> is received through the corresponding input pins and the EP[i] bit is already set. The EVTERR[i] bit is unaffected if the BP tries to set the EP[i] bit when that EP[i] bit is already set.</li> </ul>

### 41.9.6 Channel DSP Interrupt Mask (SDMABP\_INTRMASK)

Address: 20E\_C000h base + 2Ch offset = 20E\_C02Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DIMASK																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SDMABP\_INTRMASK field descriptions

Field	Description
31–0 DIMASK	The Interrupt Mask Register contains 32 interrupt generation mask bits. If bit DIMASK[i] is set, the DI[i] bit is set and an interrupt is sent to the BP when a DMA request error is detected on channel <i>i</i> (for example, EVTERR[i] is set).

### 41.9.7 DMA Request Error Register (SDMABP\_EVTERRDBG)

Address: 20E\_C000h base + 34h offset = 20E\_C034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CHNERR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SDMABP\_EVTERRDBG field descriptions

Field	Description
31–0 CHNERR	This register is the same as EVTERR except reading it does not clear its contents. This address is meant to be used in debug mode. The BP OnCE may check this register value without modifying it.

## 41.10 SDMA Internal (Core) Memory Map and Internal Register Definitions

The actual SDMA memory mapped registers are summarized in the following sections; for peripherals' memory maps, refer to the respective chapters.

The following definitions serve as a key for the SDMA internal register summary.

## SDMACORE memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20E_C000	ARM platform Channel 0 Pointer (SDMACORE_MC0PTR)	32	R	0000_0000h	<a href="#">41.10.1/2629</a>
20E_C002	Current Channel Pointer (SDMACORE_CCPtr)	32	R	0000_0000h	<a href="#">41.10.2/2629</a>
20E_C003	Current Channel Register (SDMACORE_CCR)	32	R	0000_0000h	<a href="#">41.10.3/2629</a>
20E_C004	Highest Pending Channel Register (SDMACORE_NCR)	32	R	0000_0000h	<a href="#">41.10.4/2630</a>
20E_C005	External DMA Requests Mirror (SDMACORE_EVENTS)	32	R	0000_0000h	<a href="#">41.10.5/2631</a>
20E_C006	Current Channel Priority (SDMACORE_CCPRI)	32	R	0000_0000h	<a href="#">41.10.6/2632</a>
20E_C007	Next Channel Priority (SDMACORE_NCPRI)	32	R	0000_0000h	<a href="#">41.10.7/2632</a>
20E_C009	OnCE Event Cell Counter (SDMACORE_ECOUNT)	32	R/W	0000_0000h	<a href="#">41.10.8/2633</a>
20E_C00A	OnCE Event Cell Control Register (SDMACORE_ECTL)	32	R/W	0000_0000h	<a href="#">41.10.9/2633</a>
20E_C00B	OnCE Event Address Register A (SDMACORE_EAA)	32	R/W	0000_0000h	<a href="#">41.10.10/2635</a>
20E_C00C	OnCE Event Cell Address Register B (SDMACORE_EAB)	32	R/W	0000_0000h	<a href="#">41.10.11/2635</a>
20E_C00D	OnCE Event Cell Address Mask (SDMACORE_EAM)	32	R/W	0000_0000h	<a href="#">41.10.12/2635</a>
20E_C00E	OnCE Event Cell Data Register (SDMACORE_ED)	32	R/W	0000_0000h	<a href="#">41.10.13/2636</a>
20E_C00F	OnCE Event Cell Data Mask (SDMACORE_EDM)	32	R/W	0000_0000h	<a href="#">41.10.14/2636</a>
20E_C018	OnCE Real-Time Buffer (SDMACORE_RTB)	32	R/W	0000_0000h	<a href="#">41.10.15/2637</a>
20E_C019	OnCE Trace Buffer (SDMACORE_TB)	32	R	0000_0000h	<a href="#">41.10.16/2637</a>
20E_C01A	OnCE Status (SDMACORE_OSTAT)	32	R	0000_0000h	<a href="#">41.10.17/2638</a>
20E_C01C	Channel 0 Boot Address (SDMACORE_MCHN0ADDR)	32	R	0000_0000h	<a href="#">41.10.18/2640</a>
20E_C01D	ENDIAN Status Register (SDMACORE_ENDIANNES)	32	R	0000_0001h	<a href="#">41.10.19/2641</a>
20E_C01E	Lock Status Register (SDMACORE_SDMA_LOCK)	32	R	0000_0000h	<a href="#">41.10.20/2642</a>
20E_C01F	External DMA Requests Mirror #2 (SDMACORE_EVENTS2)	32	R	0000_0000h	<a href="#">41.10.21/2643</a>



### 41.10.1 ARM platform Channel 0 Pointer (SDMACORE\_MC0PTR)

Address: 20E\_C000h base + 0h offset = 20E\_C000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MC0PTR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SDMACORE\_MC0PTR field descriptions

Field	Description
31–0 MC0PTR	Contains the address-in the ARM platform memory space-of the initial SDMA context and scripts that are loaded by the SDMA boot script running on channel 0.

### 41.10.2 Current Channel Pointer (SDMACORE\_CCPtr)

Address: 20E\_C000h base + 2h offset = 20E\_C002h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																CCPTR															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SDMACORE\_CCPtr field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–0 CCPTR	Contains the start address of the context data for the current channel: Its value is <i>CONTEXT_BASE</i> + 24* <i>CCR</i> or <i>CONTEXT_BASE</i> + 32* <i>CCR</i> where <i>CONTEXT_BASE</i> = 0x0800. The value 24 or 32 is selected according to the programmed channel scratch RAM size in the register shown in <a href="#">Channel 0 Boot Address (SDMAARM_CHN0ADDR)</a> .

### 41.10.3 Current Channel Register (SDMACORE\_CCR)

Address: 20E\_C000h base + 3h offset = 20E\_C003h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																CCR															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SDMACORE\_CCR field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4–0 CCR	Contains the number of the current running channel whose context is installed. In the case that the SDMA has finished running the channel and has entered sleep state, CCR will indicate the previous running channel. The PST bits in the OSTAT register indicate when the SDMA is in sleep state.

**41.10.4 Highest Pending Channel Register (SDMACORE\_NCR)**

Address: 20E\_C000h base + 4h offset = 20E\_C004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																NCR															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SDMACORE\_NCR field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4–0 NCR	Contains the number of the pending channel that the scheduler has selected to run next.

### 41.10.5 External DMA Requests Mirror (SDMACORE\_EVENTS)

#### NOTE

This register is very useful in the case of DMA requests that are active when a peripheral FIFO level is above the programmed watermark. The activation of the DMA request (rising edge) is detected by the SDMA logic and it can enable one or several channels. One of the channels accesses the peripheral and reads or writes a number of data that matches the watermark level (for example, if the watermark is four words, the channel reads or writes four words).

If the channel is effectively executed long after the DMA request was received, reading or writing the watermark number of data may not be sufficient to reset the DMA request (for example, if the FIFO watermark is four and at the channel execution it already contains nine pieces of data). This means no new rising edge may be detected by the SDMA, although there still remains transfers to perform. Therefore, if the channel were terminated at that time, it would not be restarted, causing potential overrun or underrun of the peripheral.

The proposed mechanism is for the channel to check this register after it has performed the "watermark" number of accesses to the peripheral. If the bit for the DMA request that triggers this channel is set, it means there is still another watermark number of data to transfer. This goes on until the bit is cleared. The same script can be used for multiple channels that require this behavior. The script can determine its channel number from the CCR register and infer the corresponding DMA request bit to check. It needs a reference table that is coherent with the request-channel matrix that the ARM platform programmed.

Address: 20E\_C000h base + 5h offset = 20E\_C005h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EVENTS																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SDMACORE\_EVENTS field descriptions**

Field	Description
31–0 EVENTS	Reflects the status of the SDMA's external DMA requests. It is meant to allow any channel to monitor the states of these SDMA inputs.  This register displays EVENTS 0-31. The EVENTS2 register displays events 32-47.

**41.10.6 Current Channel Priority (SDMACORE\_CCPRI)**

Address: 20E\_C000h base + 6h offset = 20E\_C006h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																CCPRI															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SDMACORE\_CCPRI field descriptions**

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value 0.
2–0 CCPRI	Contains the 3-bit priority of the channel whose context is installed. It is 0 when no channel is running.  <b>NOTE:</b> 1-7 current channel priority  0 no running channel

**41.10.7 Next Channel Priority (SDMACORE\_NCPRI)**

Address: 20E\_C000h base + 7h offset = 20E\_C007h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																NCPRI															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SDMACORE\_NCPRI field descriptions**

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value 0.
2–0 NCPRI	Contains the 3-bit priority of the channel the scheduler has selected to run next. It is 0 when no other channel is pending.

## 41.10.8 OnCE Event Cell Counter (SDMACORE\_ECOUNT)

Address: 20E\_C000h base + 9h offset = 20E\_C009h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																ECOUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SDMACORE\_ECOUNT field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–0 ECOUNT	The event cell counter contains the number of times minus one that an event detection must occur before generating a debug request. <ul style="list-style-type: none"> <li>This register should be written before any attempt to use the event detection counter during an event detection process.</li> <li>The counter is cleared on a JTAG reset.</li> </ul>

## 41.10.9 OnCE Event Cell Control Register (SDMACORE\_ECTL)

Address: 20E\_C000h base + Ah offset = 20E\_C00Ah

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		EN	CNT	ECTC[1:0]	DTC[1:0]	ATC[1:0]	ABTC[1:0]	AATC[1:0]	ATS[1:0]						
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SDMACORE\_ECTL field descriptions

Field	Description
31–14 Reserved	This read-only field is reserved and always has the value 0.
13 EN	Event Cell Enable. If the EN bit is set, the event cell is allowed to generate debug requests (the cell is awakened). If it is cleared, the event detection unit is disabled and no hardware breakpoint is generated, but matching conditions are still reflected on the emulation pin.  0 Cell is disabled. 1 Cell is enabled.
12 CNT	Event Counter Enable. The event counter enable bit determines if the cell counter is used during the event detection. In order to use the event counter during an event detection process, the event cell counter register should be loaded with a value equal to the number of times minus one that an event occurs before

Table continues on the next page...

**SDMACORE\_ECTL field descriptions (continued)**

Field	Description
	<p>a debug request is sent. After every event detection, the counter is decreased. When the counter reaches the value 0, the event detection cell sends a debug request to the core. The event counter register should be written and the EN bit should be set before each new event detection process uses the event counter.</p> <p>0 Counter is disabled. 1 Counter is enabled.</p>
11–10 ECTC[1:0]	<p>The event cell trigger condition bits select the combination of address and data matching conditions that generate the final address/data condition. During program execution, if this event cell trigger condition goes to 1, a debug request is sent to the SDMA. The EN bit must be set to enable the debug request generation.</p> <p>00 address ONLY 01 data ONLY 10 address AND data 11 address OR data</p>
9–8 DTC[1:0]	<p>The data trigger condition bits define when data is considered matching after comparison with the data register of the event detection unit. The operations are performed on unsigned values.</p> <p>00 equal 01 not equal 10 greater than 11 less than</p>
7–6 ATC[1:0]	<p>The address trigger condition bits select how the two address conditions (addressA and addressB) are combined to define the global address matching condition. The supported combinations are described, as follows.</p> <p>00 addressA ONLY 01 addrA AND addrB 10 addrA OR addrB 11 reserved</p>
5–4 ABTC[1:0]	<p>The Address B Trigger Condition (ABTC) controls the operations performed by address comparator B. All operations are performed on unsigned values. This comparator B outputs the addressB condition.</p> <p>00 equal 01 not equal 10 greater than 11 less than</p>
3–2 AATC[1:0]	<p>The Address A Trigger Condition (AATC) controls the operations performed by address comparator A. All operations are performed on unsigned values. This comparator A outputs the addressA condition.</p> <p>00 equal 01 not equal 10 greater than 11 less than</p>
1–0 ATS[1:0]	<p>The access type select bits define the memory access type required on the SDMA memory bus.</p> <p>00 read ONLY 01 write ONLY 10 read or write 11 -</p>

### 41.10.10 OnCE Event Address Register A (SDMACORE\_EAA)

Address: 20E\_C000h base + Bh offset = 20E\_C00Bh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																EAA															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SDMACORE\_EAA field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–0 EAA	Event Cell Address Register A computes an address A condition. It is cleared on a JTAG reset.

### 41.10.11 OnCE Event Cell Address Register B (SDMACORE\_EAB)

Address: 20E\_C000h base + Ch offset = 20E\_C00Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																EAB															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SDMACORE\_EAB field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–0 EAB	Event Cell Address Register B computes an address B condition. It is cleared on a JTAG reset.

### 41.10.12 OnCE Event Cell Address Mask (SDMACORE\_EAM)

Address: 20E\_C000h base + Dh offset = 20E\_C00Dh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																EAM															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SDMACORE\_EAM field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–0 EAM	The Event Cell Address Mask contains a user-defined address mask value. This mask is applied to the address value latched from the memory address bus before performing the address comparison.  <b>NOTE:</b> There is a common address mask value for both address comparators. If bit <i>i</i> of this register is set, then bit <i>i</i> of the address value latched from the memory bus does not influence the result of the address comparison. The register is cleared on a JTAG reset.

**41.10.13 OnCE Event Cell Data Register (SDMACORE\_ED)**

Address: 20E\_C000h base + Eh offset = 20E\_C00Eh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SDMACORE\_ED field descriptions**

Field	Description
31–0 ED	The event cell data register contains a user defined data value. This data value is an input for the data comparator which generates the data condition. It is cleared on a JTAG reset.

**41.10.14 OnCE Event Cell Data Mask (SDMACORE\_EDM)**

Address: 20E\_C000h base + Fh offset = 20E\_C00Fh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EDM																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SDMACORE\_EDM field descriptions**

Field	Description
31–0 EDM	The event cell data mask register contains the user-defined data mask value. <ul style="list-style-type: none"> <li>This mask is applied to the data value latched from the memory bus before performing the data comparison.</li> <li>Setting bit <i>i</i> of the event cell data mask register means that bit <i>i</i> of the data value latched from the address bus does not influence the result of the data comparison.</li> <li>The data mask is cleared on a JTAG reset.</li> </ul>



### 41.10.15 OnCE Real-Time Buffer (SDMACORE\_RTb)

Address: 20E\_C000h base + 18h offset = 20E\_C018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RTB																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SDMACORE\_RTb field descriptions

Field	Description
31–0 RTB	<p>The Real Time Buffer register stores and retrieves run time information without putting the SDMA in debug mode. Writing to that register triggers a pulse on a specific real-time debug pin whose connection depends on the chip implementation.</p> <p>The RTB value can be accessed by the OnCE under ARM platform or JTAG control using the rbuffer command.</p>

### 41.10.16 OnCE Trace Buffer (SDMACORE\_TB)

Address: 20E\_C000h base + 19h offset = 20E\_C019h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			TBF	TADDR											
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TADDR		CHFADDR													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### SDMACORE\_TB field descriptions

Field	Description
31–29 Reserved	This read-only field is reserved and always has the value 0.
28 TBF	<p>The Trace Buffer Flag is set when the buffer contains the addresses of a valid change of flow. The contents of the buffer should be ignored otherwise.</p> <p>0 Invalid information 1 Valid information</p>
27–14 TADDR	The target address is the address taken after the execution of the change of flow instruction.
13–0 CHFADDR	The change of flow address is the address where the change of flow is taken when executing a change of flow instruction.

## 41.10.17 OnCE Status (SDMACORE\_OSTAT)

Address: 20E\_C000h base + 1Ah offset = 20E\_C01Ah

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PST[3:0]				RCV	EDR	ODR	SWB	MST	0				ECDR[2:0]		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SDMACORE\_OSTAT field descriptions

Field	Description																												
31–16 Reserved	This read-only field is reserved and always has the value 0.																												
15–12 PST[3:0]	<p>The Processor Status bits reflect the state of the SDMA RISC engine.</p> <ul style="list-style-type: none"> <li>The "Program" state is the usual instruction execution cycle.</li> <li>The "Data" state is inserted when there are wait-states during a load or a store on the data bus (ld or st).</li> <li>The "Change of Flow" state is the second cycle of any instruction that breaks the sequence of instructions (jumps and channel-switching instructions).</li> <li>The "Change of Flow in Loop" state is used when an error causes a hardware loop exit.</li> <li>The "Debug" state means the SDMA is in debug mode.</li> <li>The "Functional Unit" state is inserted when there are wait-states during a load or a store on the functional units bus (ldf or stf).</li> <li>In "Sleep" modes, no script is running (this is the RISC engine idle state). The "after Reset" is slightly different because no context restoring phase will happen when a channel is triggered: The script located at address 0 will be executed (boot operation).</li> <li>The "in Sleep" states are the same as above except they do not have any corresponding channel. They are used when entering debug mode after reset; the reason is that it is necessary to return to the "Sleep after Reset" state when leaving debug mode.</li> </ul> <table> <tr><td>0</td><td>Program</td></tr> <tr><td>1</td><td>Data</td></tr> <tr><td>2</td><td>Change of Flow</td></tr> <tr><td>3</td><td>Change of Flow in Loop</td></tr> <tr><td>4</td><td>Debug</td></tr> <tr><td>5</td><td>Functional Unit</td></tr> <tr><td>6</td><td>Sleep</td></tr> <tr><td>7</td><td>Save</td></tr> <tr><td>8</td><td>Program in Sleep</td></tr> <tr><td>9</td><td>Data in Sleep</td></tr> <tr><td>10</td><td>Change of Flow in Sleep</td></tr> <tr><td>11</td><td>Change Flow Loop Sleep</td></tr> <tr><td>12</td><td>Debug in Sleep</td></tr> <tr><td>13</td><td>Functional Unit in Sleep</td></tr> </table>	0	Program	1	Data	2	Change of Flow	3	Change of Flow in Loop	4	Debug	5	Functional Unit	6	Sleep	7	Save	8	Program in Sleep	9	Data in Sleep	10	Change of Flow in Sleep	11	Change Flow Loop Sleep	12	Debug in Sleep	13	Functional Unit in Sleep
0	Program																												
1	Data																												
2	Change of Flow																												
3	Change of Flow in Loop																												
4	Debug																												
5	Functional Unit																												
6	Sleep																												
7	Save																												
8	Program in Sleep																												
9	Data in Sleep																												
10	Change of Flow in Sleep																												
11	Change Flow Loop Sleep																												
12	Debug in Sleep																												
13	Functional Unit in Sleep																												

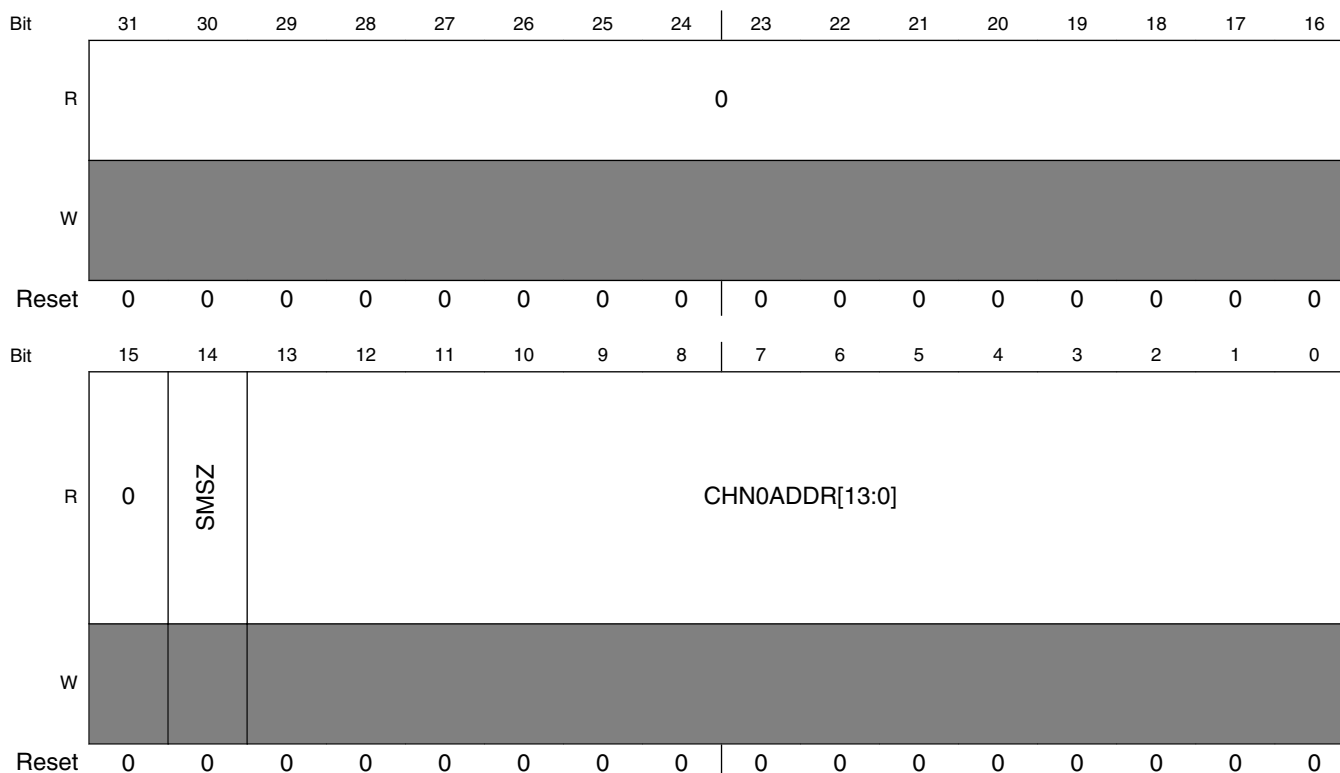
Table continues on the next page...

**SDMACORE\_OSTAT field descriptions (continued)**

Field	Description
	14 Sleep after Reset 15 Restore
11 RCV	After each write access to the real time buffer (RTB), the RCV bit is set. This bit is cleared after execution of an <code>rbuffer</code> command and on a JTAG reset.
10 EDR	This flag is raised when the SDMA has entered debug mode after an external debug request.
9 ODR	This flag is raised when the SDMA has entered debug mode after a OnCE debug request.
8 SWB	This flag is raised when the SDMA has entered debug mode after a software breakpoint.
7 MST	This flag is raised when the OnCE is controlled from the ARM platform peripheral interface. 0 JTAG interface controls the OnCE. 1 ARM platform peripheral interface controls the OnCE.
6–3 Reserved	This read-only field is reserved and always has the value 0.
2–0 ECDR[2:0]	Event Cell Debug Request. If the debug request comes from the event cell, the reason for entering debug mode is given by the EDR bits. The encoding of the EDR bits is useful to find out more precisely why the debug request was generated. A debug request from an event cell is generated for a specific combination of the addressA, addressB, and data conditions; the value of those fields is given by the EDR bits. If all three bits of the EDR are reset, then it did not generate any debug request. If the cell did generate a debug request, then at least one EDR bit is set; the meaning of the encoding is as follows:  0 1 matched addressA condition 1 1 matched addressB condition 2 1 matched data condition

## 41.10.18 Channel 0 Boot Address (SDMACORE\_MCHN0ADDR)

Address: 20E\_C000h base + 1Ch offset = 20E\_C01Ch



### SDMACORE\_MCHN0ADDR field descriptions

Field	Description
31–15 Reserved	This read-only field is reserved and always has the value 0.
14 SMSZ	<p>The bit 14 (Scratch Memory Size) determines if scratch memory must be available after every channel context. After reset, it is equal to 0, which defines a RAM space of 24 words for each channel. All of this area stores the channel context. By setting this bit, 32 words are reserved for every channel context, which gives eight additional words that can be used by the channel script to store any type of data. Those words are never erased by the context switching mechanism.</p> <p>0 24 words per context 1 32 words per context</p>
13–0 CHN0ADDR[13:0]	Contains the address of the channel 0 routine programmed by the ARM platform; it is loaded into a general register at the very start of the boot and the SDMA jumps to the address it contains. By default, it points to the standard boot routine in ROM.

## 41.10.19 ENDIAN Status Register (SDMACORE\_ENDIANNES)

Address: 20E\_C000h base + 1Dh offset = 20E\_C01Dh

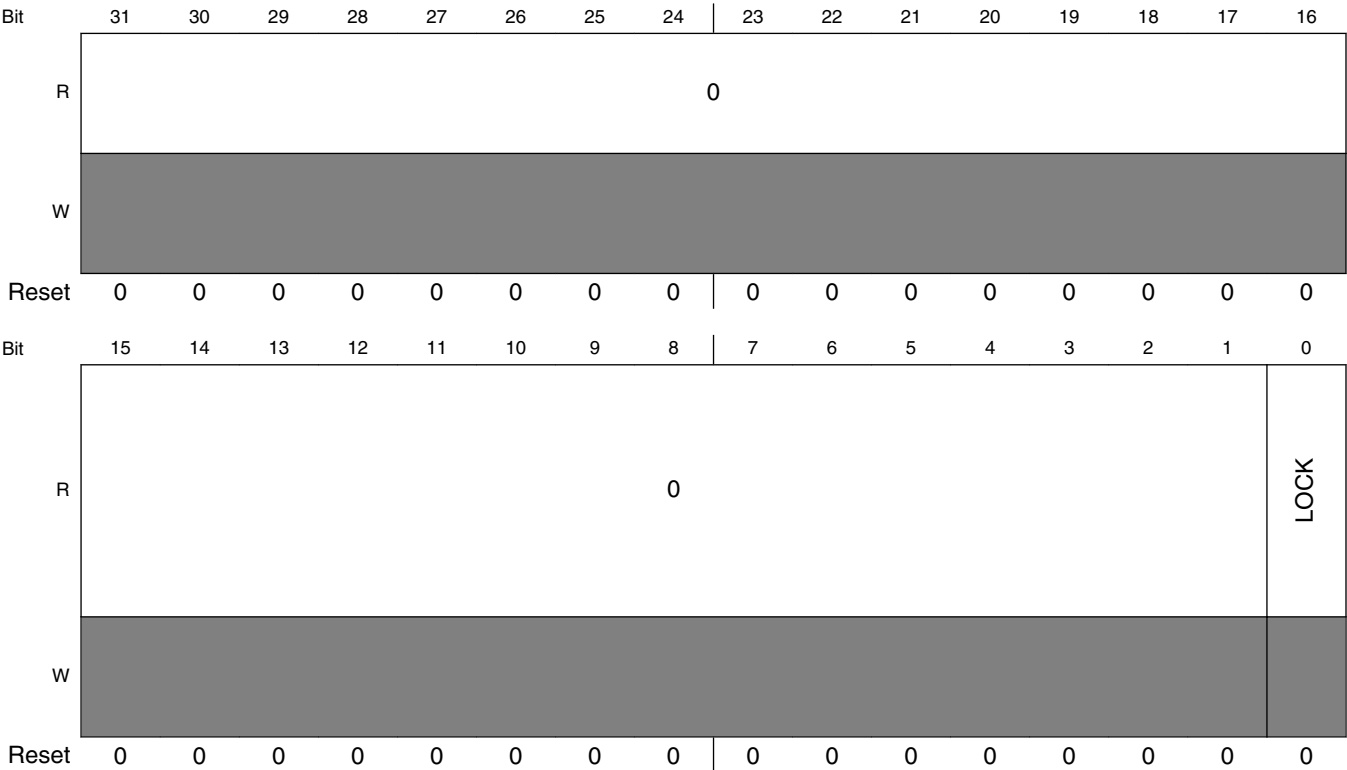
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0												0		APEND	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

### SDMACORE\_ENDIANNES field descriptions

Field	Description
31–3 Reserved	This read-only field is reserved and always has the value 0.
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 APEND	APEND indicates the endian mode of the Peripheral and Burst DMA interfaces. This bit is tied to logic '1' indicating little-endian mode.  0 - ARM platform is in big-endian mode 1 - ARM platform is in little-endian mode

41.10.20 Lock Status Register (SDMACORE\_SDMA\_LOCK)

Address: 20E\_C000h base + 1Eh offset = 20E\_C01Eh



SDMACORE\_SDMA\_LOCK field descriptions

Field	Description
31–1 Reserved	This read-only field is reserved and always has the value 0.
0 LOCK	The LOCK bit reports the value of the LOCK bit in the SDMA_LOCK status register. SDMA software may use this value to determine if certain operations such as loading of new scripts is allowed.  0 - LOCK bit clear 1 - LOCK bit set

### 41.10.21 External DMA Requests Mirror #2 (SDMACORE\_EVENTS2)

Address: 20E\_C000h base + 1Fh offset = 20E\_C01Fh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																EVENTS															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SDMACORE\_EVENTS2 field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–0 EVENTS	Reflects the status of the SDMA's external DMA requests. It is meant to allow any channel to monitor the states of these SDMA inputs.  This register displays EVENTS 32-47. The separate EVENTS register displays events 0-31.

### 41.11 SDMA Peripheral Registers

Refer to the respective peripherals' chapters for more information.





# Chapter 42

## SiPix Display Controller (SPDC)

### 42.1 Overview

This chapter describes the detailed architecture of the SiPix Display Controller (SPDC) for E-Book application. It provides a detailed description of the SPDC for digital design and software development.

The SPDC provides control signals for the source driver and gate drivers. This IP provides a high performance, low cost solution for SiPix EPDs (Electronic Paper Display). Partial update and concurrent display updates resulting in high responsive screen changes are also implemented for these applications. The SPDC module is defined to co-work in conjunction with the eXPX IP module to form a complete display processing solution, such as rotation and flip function.

#### 42.1.1 Features

The key features of the SPDC are as follows:

- Support 22 resolutions:
  - 800x600, 600x800, 1024x768, 768x1024, 1024x800, 800x1024, 1024x600, 600x1024, 1200x825, 825x1200, 1280x825, 825x1280, 1280x800, 800x1280, 1280x768, 768x1280, 1280x960, 960x1280, 1280x1024, 1024x 1280, 1600x1200, 1200x1600
- Support driver IC:
  - Source driver: K7600
  - Gate driver: MEXI2300, AUO-G5901
  - Source driver + Gate driver: K7900, K7902, K7903
- Industry standard bus interfaces (AMBA AHB and APB)
- Low-power mode operation via architectural clock gating
- Support full and partial update mode

### 42.1.2 Block Diagram

The top-level view of the SPDC is shown in figure below.

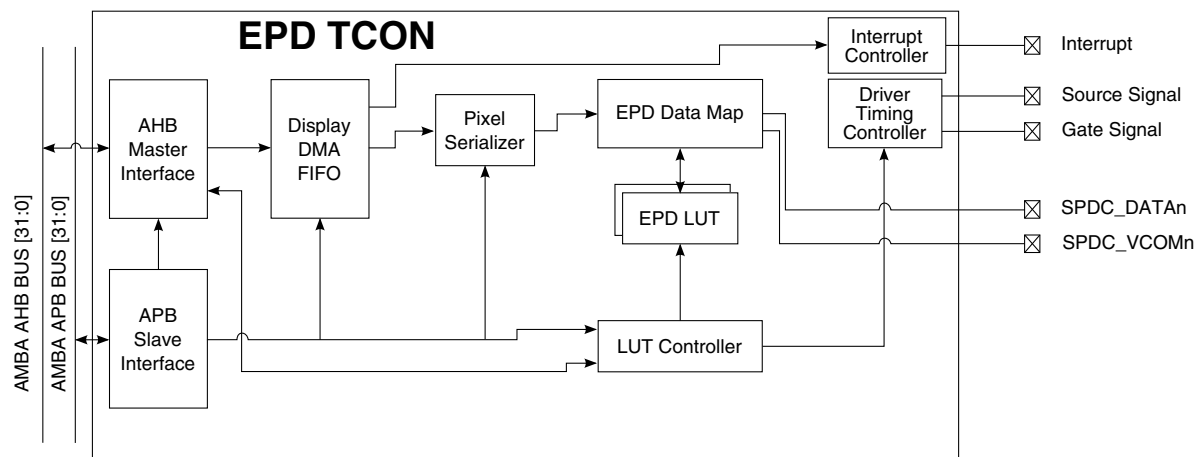


Figure 42-1. Overall Block Diagram of SPDC

The SPDC can be separated into three major groups. The first group, including the AHB Master Interface, the APB Slave Interface, and the Display DMA FIFO, is responsible for accessing the image pixel data from DRAM memory and receives the APB register setting from CPU. The second group, including Pixel Serializer, EPD Data Map, EPD LUT, and LUT Controller, is responsible for using the image pixel data to look up waveform LUT to generate driving waveform of SiPix EPD panel. The last group includes Driver Timing Controller and Interrupt Controller, and is responsible for generating the driver driving timing and issuing all kind of interrupt signals.

### 42.2 External Signals

Table 42-1. SPDC I/O Port

Function.	Port Name	I/O	Width	Description
Driver Control/Data Signal	SPDC_CL	O	1	Driver Clock
	SPDC_XDIOR	O	1	Source driver data right start pulse
	SPDC_XDIOL	O	1	Source driver data left start pulse
	SPDC_LD	O	1	Source driver load signal (latch data to DAC)

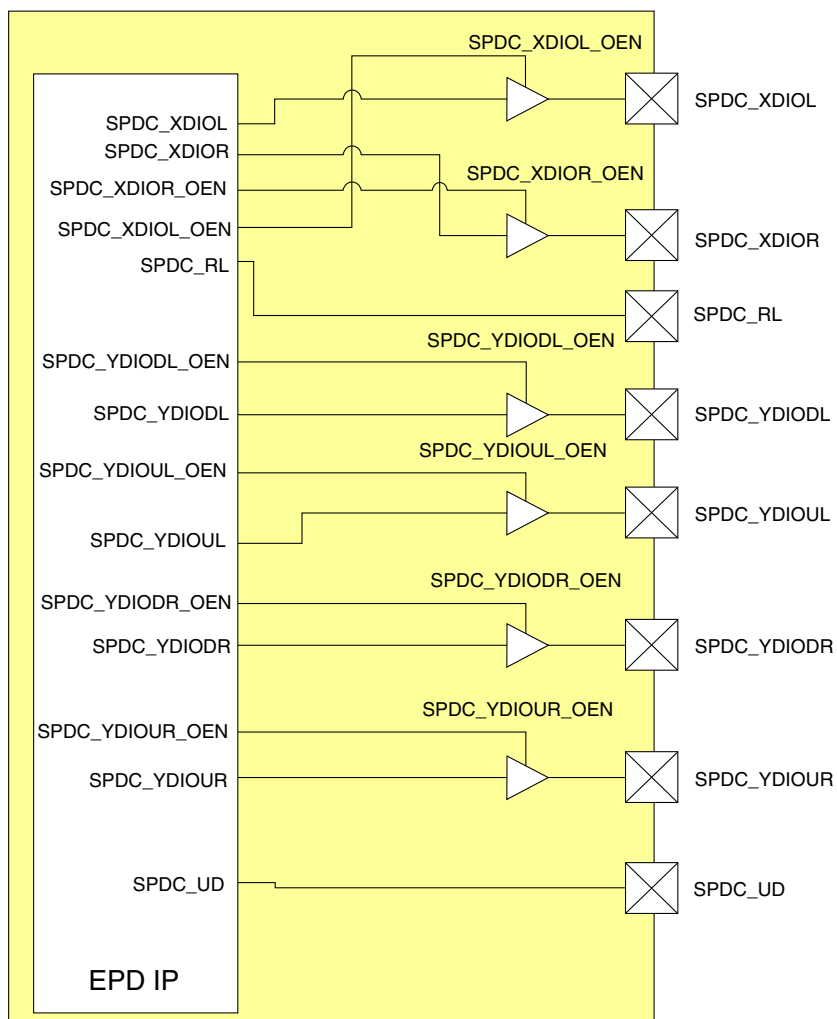
Table continues on the next page...

**Table 42-1. SPDC I/O Port (continued)**

	SPDC_RL (SOE)	O	1	Panel scan direction [0: Right to Left, 1: Left to Right] (could be setting to SOE pin - Source driver output enable)
	SPDC_YDIODL	O	1	Left gate driver down start pulse
	SPDC_YDIOUL	O	1	Left gate driver up start pulse
	SPDC_YDIOUR	O	1	Right gate driver down start pulse
	SPDC_YDIOUR	O	1	Right gate driver up start pulse
	SPDC_YOEL	O	1	Left gate driver output enable
	SPDC_SOE (YOER)	O	1	Source driver output enable (Can be set to YOER pin - Right gate driver output enable)
	SPDC_YCKL	O	1	Left gate driver clock
	SPDC_YCKR	O	1	Right gate driver clock
	SPDC_UD (SOE)	O	1	Gate scan direction [0: Up to Down, 1: Down to Up] (Can be set to SOE pin - Source driver output enable)
	SPDC_VCOM	O	2	SPDC_VCOM selection
	SPDC_DATA	O	16	Driver data

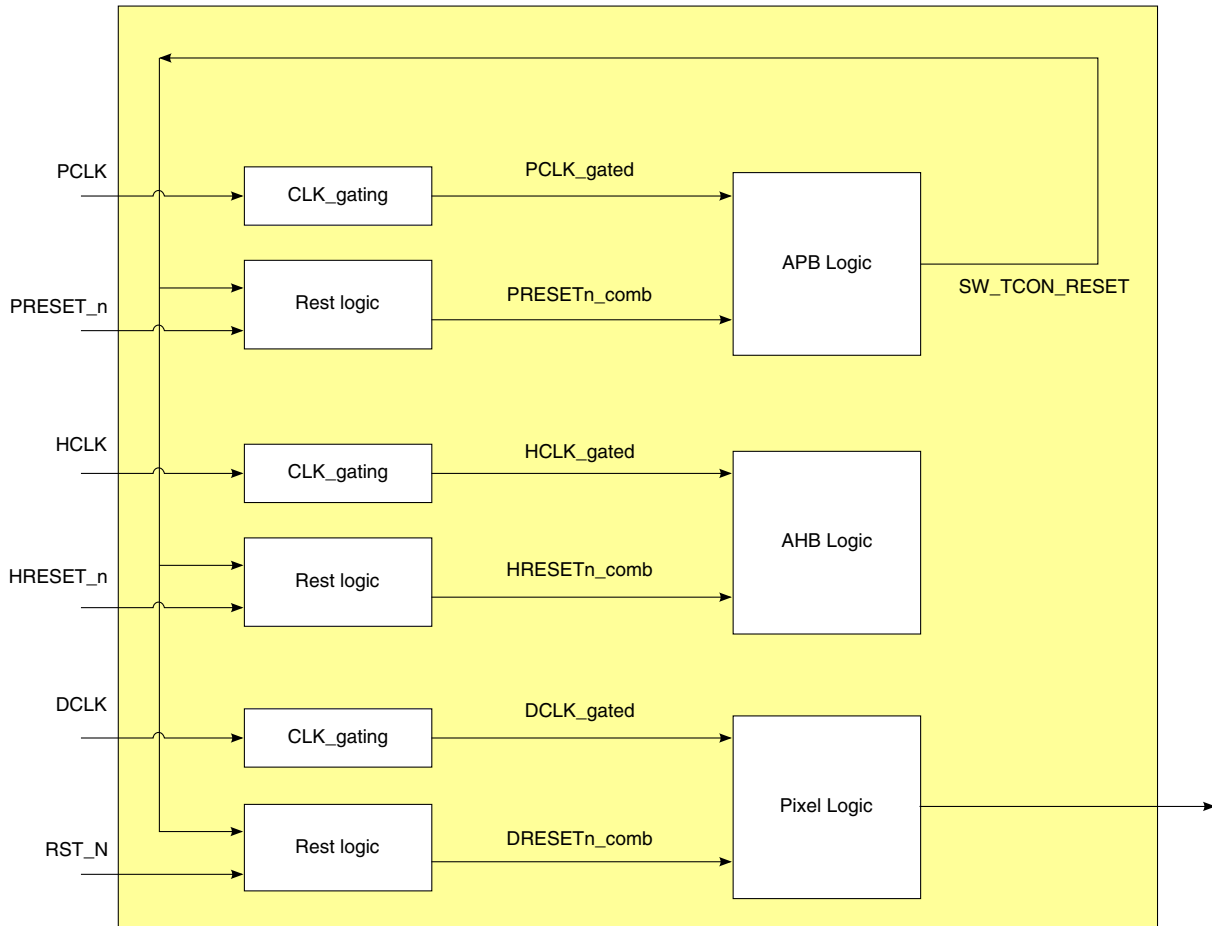
## 42.2.1 Chip Level Connection

### 42.2.1.1 Source/Gate Start Pulse (XDIO/YDIO) Connection



**Figure 42-2. XDIO/YDIO Signal Connection**

### 42.2.1.2 Clock and Reset Signal Connection



**Figure 42-3. Clock and Reset Signal Connection**

### 42.2.2 Chip Level Pinout List

Table found here shows the pinout list and pinout type of SPDC.

**Table 42-2. SPDC Pinout List**

Pad #	Name	output	output enable (Low Active)	pad type
1	SPDC_CL	SPDC_CL	0	Type 1
2	SPDC_XDIOR	SPDC_XDIOR	XDIOR_OEN	Type 6
3	SPDC_XDIOL	SPDC_XDIOL	XDIOL_OEN	Type 6
4	SPDC_LD	SPDC_LD	0	Type 1

*Table continues on the next page...*

**Table 42-2. SPDC Pinout List  
(continued)**

5	SPDC_RL	SPDC_RL	0	Type 1
6	SPDC_YDIODL	SPDC_YDIODL	SPDC_YDIODL_OEN	Type 6
7	SPDC_YDIOUL	SPDC_YDIOUL	SPDC_YDIOUL_OEN	Type 6
8	SPDC_YDIODR	SPDC_YDIODR	SPDC_YDIODR_OEN	Type 6
9	SPDC_YDIOUR	SPDC_YDIOUR	SPDC_YDIOUR_OEN	Type 6
10	SPDC_YOEL	SPDC_YOEL	0	Type 1
11	SPDC_YOER	SPDC_YOER	0	Type 1
12	SPDC_YCKL	SPDC_YCKL	0	Type 1
13	SPDC_YCKR	SPDC_YCKR	0	Type 1
14	SPDC_UD	SPDC_UD	0	Type 1
15	SPDC_VCOM1	SPDC_VCOM1	0	Type 1
16	SPDC_VCOM0	SPDC_VCOM0	0	Type 1
17	SPDC_DATA15	SPDC_DATA15	0	Type 1
18	SPDC_DATA14	SPDC_DATA14	0	Type 1
19	SPDC_DATA13	SPDC_DATA13	0	Type 1
20	SPDC_DATA12	SPDC_DATA12	0	Type 1
21	SPDC_DATA11	SPDC_DATA11	0	Type 1
22	SPDC_DATA10	SPDC_DATA10	0	Type 1
23	SPDC_DATA09	SPDC_DATA09	0	Type 1
24	SPDC_DATA08	SPDC_DATA08	0	Type 1
25	SPDC_DATA07	SPDC_DATA07	0	Type 1
26	SPDC_DATA06	SPDC_DATA06	0	Type 1
27	SPDC_DATA05	SPDC_DATA05	0	Type 1
28	SPDC_DATA04	SPDC_DATA04	0	Type 1
29	SPDC_DATA03	SPDC_DATA03	0	Type 1
30	SPDC_DATA02	SPDC_DATA02	0	Type 1
31	SPDC_DATA01	SPDC_DATA01	0	Type 1
32	SPDC_DATA00	SPDC_DATA00	0	Type 1

Figure below shows the detailed circuit of each pinout type.

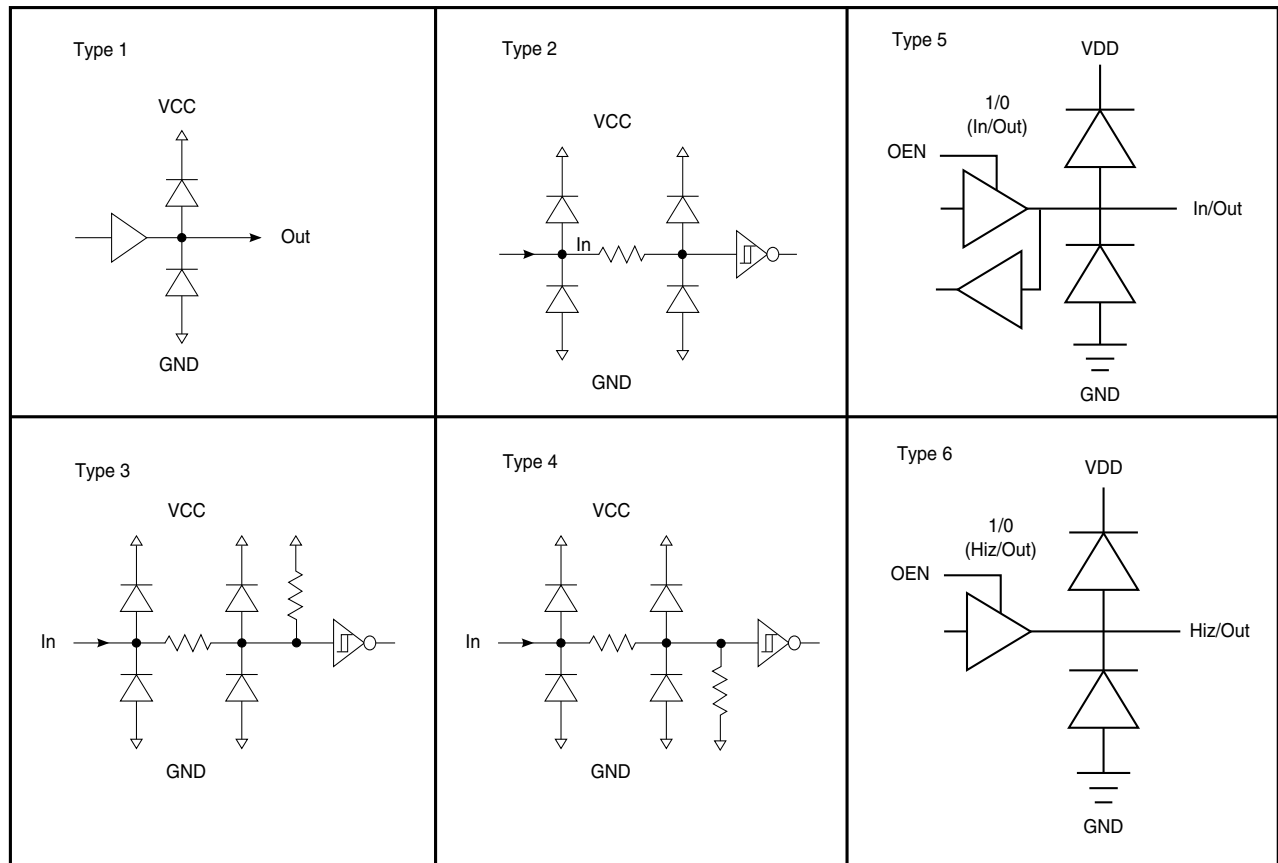
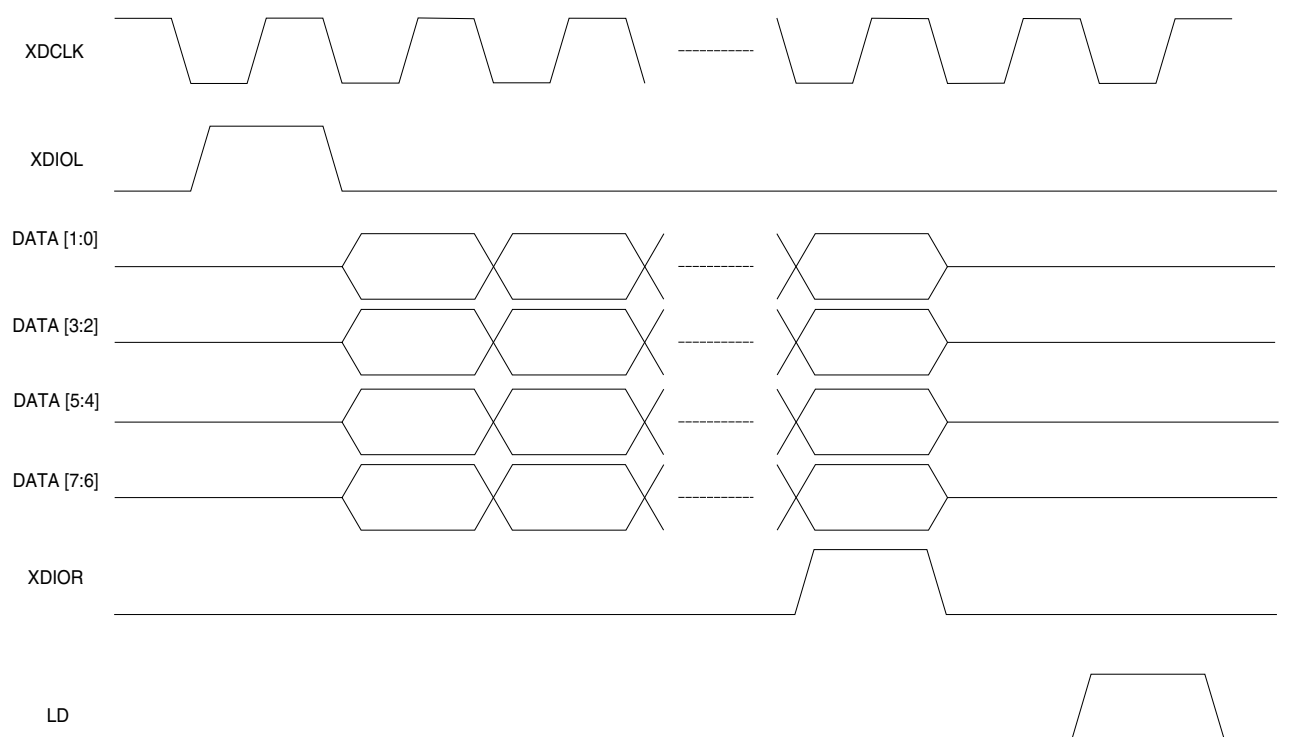


Figure 42-4. Detailed Circuit of Each Pinout Type (Pull Up/Pull Down Resistors 40K)

### 42.2.3 Display Function Timing Description

### 42.2.3.1 Source Driver Timing



**Figure 42-5. SPDC Source Driver Timing Diagram**



### 42.2.3.2 Gate Driver Timing

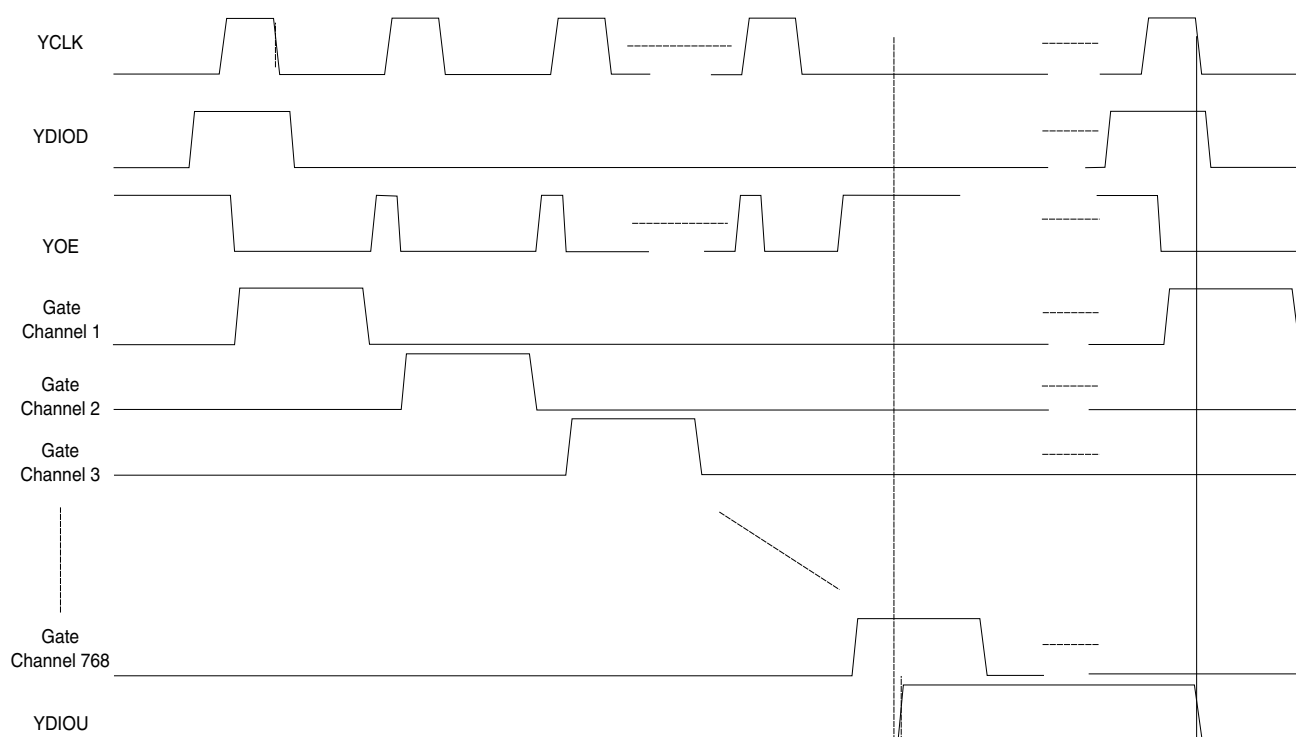


Figure 42-6. SPDC Gate Driver Timing Diagram

## 42.3 Software Programming Model

This section intends to introduce the details of driving an SPDC IP using software.

### 42.3.1 SPDC Initial Setup Programming

Before starting to drive the EPD panel using SPDC IP, the system settings need to be initialized by the software. The SPDC initial setup flow is described as follows:

1. Step 1 - Configure the SPDC's working memory in DDR memory from kernel space. For a detailed description, see [Working Memory and Bus Bandwidth Requirements](#).
2. Step 2 - Move the waveform LUT from flash to DDR memory. For a detailed description, see [Reading SiPix EPD Driving Waveform LUT](#).
3. Step 3 - Driving the driver IC power sequence. For a detailed description, see [SiPix EPD Panel Power Sequence Control](#).
4. Step 4 - Configure the SPDC system clock. For a detailed description, see [SPDC System Clock Configuration](#).

5. Step 5 - Turn off the clock gating function (low power mode). For a detailed description, see [Clock Gating Architecture Configuration](#).
6. Step 6 - Configure the interrupt setting. For a detailed description, see [Interrupts Configuration](#).
7. Step 7 - Send some basic initial setting to SPDC, such as TCON initial setting and working frame buffer address registers. For a detailed description, see [Initial Configuration Command Sequence](#).

## 42.3.1.1 Working Memory and Bus Bandwidth Requirements

### 42.3.1.1.1 SPDC Working Memory Buffer

Five kinds of working memory buffers will be accessed by SPDC during panel screen updating at most. They are all expected to reside in system memory, such as external DDR memory, SDRAM, DRAM. And each has particular requirements for gray and color (RGBW) mode:

#### 42.3.1.1.1.1 GRAY MODE

- Next Frame Memory Buffer -The next frame memory buffer, which is pointed by SPDC\_NEXT\_BUF and it must have Panel Resolution (Height x Width) / 2 bytes allocation in system memory (WORD-ALIGN). This buffer is only used to store the next image pixel data. That is, end user put the image data which want to be shown on panel in this memory. The figure below shows the pixel data arrangement in next frame memory buffer.

Bit	B[31:28]	B[27:24]	B[23:20]	B[19:16]	B[15:12]	B[11:8]	B[7:4]	B[3:0]
Pixel	P8	P7	P6	P5	P4	P3	P2	P1

**Figure 42-7. Pixel Data Arrangement in Next Frame Memory Buffer**

- Current Frame Memory Buffer -The current frame memory buffer, which is pointed by SPDC\_CURRENT\_BUF and it must have Panel Resolution (Height x Width) / 2 bytes allocation in system memory (WORD-ALIGN). This buffer is only used to store the current image pixel data. (The content is showing on panel)
- Previous Frame Memory Buffer -The previous frame memory buffer, which is pointed by SPDC\_PREVIOUS\_BUF and it must have Panel Resolution (Height x Width) / 2 bytes allocation in system memory (WORD-ALIGN). This buffer is only used to store the previous image pixel data. (The content had shown on panel)
- Counter Frame Memory Buffer - The counter frame memory buffer, which is pointed by SPDC\_FRM\_CNT\_BUF and it must have Panel Resolution (Height x Width)

bytes allocation in system memory (WORD-ALIGN). This buffer is only used to record the frame counter data of each pixel in mode 4 (High Speed Handwriting Mode).

- **LUT Memory Buffer** - The waveform LUT memory buffer, which is pointed by SPDC\_LUT\_BUF and it must have 2 Mbytes allocation in system memory (WORD-ALIGN). This buffer is used to store the SPDC driving waveform data. The figure below shows the LUT waveform data arrangement in LUT memory buffer.

Bit	B[31:24]	B[23:16]	B[15:8]	B[7:0]
Waveform Data	Data 4	Data 3	Data 2	Data 1

**Figure 42-8. LUT Waveform Data Arrangement in LUT Memory Buffer**

The total working memory requirements in gray mode are given as follows (in bytes):

- $(\text{Panel Resolution (Height x Width)} \times 2.5) + 2 \text{ M (bytes)}$

#### 42.3.1.1.1.2 COLOR MODE

In color mode, it can not use the mode 4 (High Speed Handwriting mode) to update panel screen. So it didn't need to set the counter frame memory buffer.

- **Next Frame Memory Buffer** -The next frame memory buffer, which is pointed by SPDC\_NEXT\_BUF and it must have Panel Resolution (Height x Width) x 2 bytes allocation in system memory (WORD-ALIGN). This buffer is only used to store the next image pixel data. That is, the end user put the image data which want to be shown on the panel in this memory. The Figure 45-7 shows the pixel data arrangement in next frame memory buffer.

Bit	B[31:28]	B[27:24]	B[23:20]	B[19:16]	B[15:12]	B[11:8]	B[7:4]	B[3:0]
Pixel	R2	G2	B2	W2	R1	G1	B1	W1

**Figure 42-9. Pixel Data Arrangement in Next Frame Memory Buffer**

- **Current Frame Memory Buffer** -The current frame memory buffer, which is pointed by SPDC\_CURRENT\_BUF and it must have Panel Resolution (Height x Width) x 2 bytes allocation in system memory (WORD-ALIGN). This buffer is only used to store the current image pixel data. (The content is showing on panel)
- **Previous Frame Memory Buffer** -The previous frame memory buffer, which is pointed by SPDC\_PREVIOUS\_BUF and it must have Panel Resolution (Height x

Width) x 2 bytes allocation in system memory (WORD-ALIGN). This buffer is only used to store the previous image pixel data. (The content had shown on panel)

- LUT Memory Buffer - The waveform LUT memory buffer, which is pointed by SPDC\_LUT\_BUF and it must have 2 Mbytes allocation in system memory (WORD-ALIGN). This buffer is used to store the SPDC driving waveform data. The Figure 45-6 shows the LUT waveform data arrangement in LUT memory buffer.

In summary, the total working memory requirements in color mode are given as follows (in bytes):

- (Panel Resolution (Height x Width) x 6 ) + 2 M (bytes)

### NOTE

Software designer should always operate with next frame memory buffer, and the SPDC IP will update current/previous/counter frame memory buffer automatically.

#### 42.3.1.1.2 Bus Bandwidth Calculation

The maximum bandwidth scenario involves processing mode 4 (High Speed Handwriting Mode) while end user is performing handwriting operations. The mode 4 bandwidth could be calculated as follows:

- Current Frame Memory - Panel Resolution (Height x Width) x Frame Rate (50 Hz) x Per Pixel Size (0.5 bytes)
- Previous Frame Memory - Panel Resolution (Height x Width) x Frame Rate (50 Hz) x Per Pixel Size (0.5 bytes)
- Counter Frame Memory - Panel Resolution (Height x Width) x Frame Rate (50 Hz) x Per Pixel Size (1 bytes) x Concurrently Read and Write (2)

In summary, the total maximum bandwidth requirements in mode 4 are given as follows (in bytes):

- Panel Resolution (Height x Width) x Frame Rate (50 Hz) x 3 (bytes). For example: when 1600 x 1200 resolution in mode 4, the maximum bandwidth requirement is 1600 x 1200 x 50 x 3 (bytes) = 288 M (bytes)

#### 42.3.1.2 Reading SiPix EPD Driving Waveform LUT

The system should move the LUT from flash to system memory, and then system needs to record the LUT start address in system memory.

Because the system need to send it to SPDC\_LUT\_BUF register when SPDC initial setup flow during Step 7.

### 42.3.1.3 SiPix EPD Panel Power Sequence Control

It is assumed that the driver layers will control the driver IC power on/off sequence by the GPIO interface.

The maximum GPIO pins for power on/off sequence are total need 10 pins.

The following figure shows the latest power on/off sequence and the software designer should refer to this figure to control the SiPix EPD panel power.

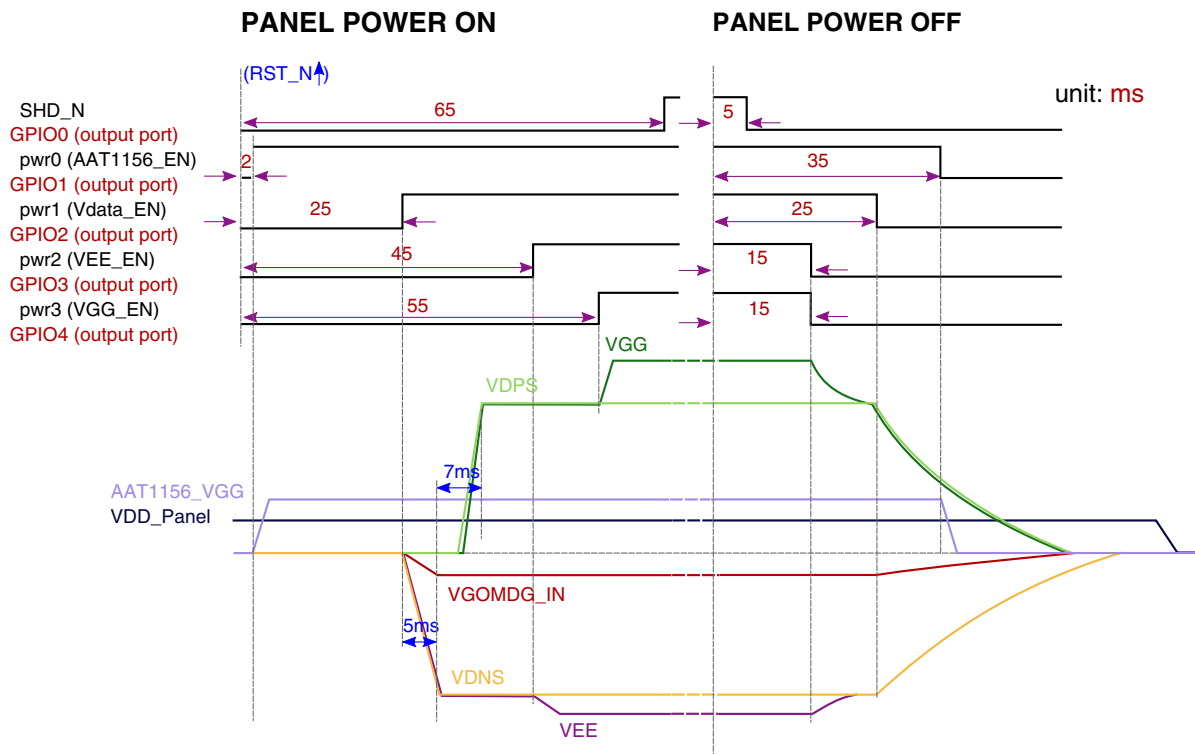


Figure 42-10. Latest Power On/Off Sequence for K7902 Driver IC

#### NOTE

The driver IC VDD should be turned on before analog power. The power on/off sequence may have changed with different versions of the power board.

### 42.3.1.4 SPDC System Clock Configuration

The SPDC supports a number of resolutions. Each resolution forces a particular SPDC systems clock rate (DCLK).

Table 42-3 shows default blanking timing setting of SPDC in each resolution and Table 42-4 shows the minimum blanking timing setting of SPDC in each resolution.

**Table 42-3. System clock required in each resolution (for default blanking timing)**

Resolution				Ideal DCLK Frequency (MHz)	Ideal Frame Rate (Hz)
H	V	H_Blanking	V_Blanking		
800	600	520	10	40.26	50
600	800	400	10	40.50	50
1024	600	300	10	40.38	50
600	1024	180	10	40.33	50
1024	768	20	10	40.61	50
768	1024	25	10	41.00	50
1024	800	10	10	41.88	50
800	1024	10	10	41.88	50
1280	768	50	10	51.74	50
768	1280	50	10	52.76	50
1200	825	50	10	52.19	50
825	1200	50	10	52.94	50
1280	800	50	10	53.87	50
800	1280	50	10	54.83	50
1280	825	50	10	55.53	50
825	1280	50	10	56.44	50
1280	960	50	10	64.51	50
960	1280	50	10	65.15	50
1280	1024	50	10	68.76	50
1024	1280	50	10	69.27	50
1600	1200	50	10	99.83	50
1200	1600	50	10	100.63	50

**Table 42-4. System clock required in each resolution (for minimum blanking timing)**

Resolution				Ideal DCLK Frequency (MHz)	Ideal Frame Rate (Hz)
H	V	H_Blanking	V_Blanking		
800	600	10	10	24.71	50
600	800	10	10	24.71	50
1024	600	10	10	31.54	50
600	1024	10	10	31.54	50
1024	768	10	10	40.22	50

*Table continues on the next page...*

**Table 42-4. System clock required in each resolution (for minimum blanking timing)  
(continued)**

768	1024	10	10	40.22	50
1024	800	10	10	41.88	50
800	1024	10	10	41.88	50
1280	768	10	10	50.18	50
768	1280	10	10	50.18	50
1200	825	10	10	50.52	50
825	1200	10	10	50.52	50
1280	800	10	10	52.25	50
800	1280	10	10	52.25	50
1280	825	10	10	53.86	50
825	1280	10	10	53.86	50
1280	960	10	10	62.57	50
960	1280	10	10	62.57	50
1280	1024	10	10	66.69	50
1024	1280	10	10	66.69	50
1600	1200	10	10	97.41	50
1200	1600	10	10	97.41	50

### 42.3.1.5 Clock Gating Architecture Configuration

The SPDC includes hardware controlled automatic clock gating throughout the design.

This clock gating function will gate off all clocks of sequential elements (flip-flops) when the display content is not being updated. This tier of clock gating is useful for reducing power consumption. The SPDC provides clock-gating feature that could be controlled by the software designer.

Because the SPDC hardware default setting will automatic gated-off all clocks that the software designer need to send SPDC\_SW\_GATE\_CLK setting to clear the SPDC\_SW\_GATE\_CLK[GATING\_ALL\_CLK\_EN] field at initial being to use SPDC.

If the application has finished all updates, the software designer can re-enable the gated-off all clocks function by setting the SPDC\_SW\_GATE\_CLK[GATING\_ALL\_CLK\_EN] field to save power.

#### NOTE

All screen updates must be complete before the software designer can place the SPDC into low power mode.

When the application is ready to send more screen updates, the software designer can immediately re-enable the SPDC by clearing the `SPDC_SW_GATE_CLK[GATING_ALL_CLK_EN]` field to un-gating all clocks. This un-gating of all clocks is immediate and the software designer is free to send update requests to SPDC.

### 42.3.1.6 Interrupts Configuration

The SPDC has a single interrupt signal intended to be connected to a system interrupt controller.

The interrupt will be asserted by SPDC as long as an unmasked interrupt source is active.

#### 42.3.1.6.1 Interrupt Sources

The SPDC has four kinds of unique interrupt sources. Each interrupt source has a particular interrupt status bit which can be read in the `SPDC_INT_ST_CLR` register. When an interrupt event happens, the appropriate bit within the `SPDC_INT_ST_CLR` register is automatically asserted by SPDC. Each of the bits in `SPDC_INT_ST_CLR` register is first AND'ed with its enable bit (in `SPDC_INT_EN`), and then all the masked bits are logically OR'ed together to generate the final single interrupt signal output to the system interrupt controller.

#### 42.3.1.6.2 Enable and Disable Interrupts

Each of the SPDC interrupt sources can be individually enable through the bits in the `SPDC_INT_EN` register. By the way, if an interrupt source is masked out (`SPDC_INT_EN` bit set to 0), its status will not be available in the `SPDC_INT_ST_CLR` register and the interrupt signal will not be asserted.

#### 42.3.1.6.3 Handle and Clear Interrupts

When the software interrupt service routine is entered as a result of a SPDC interrupt, it should read the `SPDC_EPD_INT_ST_CLR` interrupt status register. Once the interrupt has been handled, the software designer should clear the interrupt signal by writing to `SPDC_EPD_INT_ST_CLR` register (set the corresponding interrupt bit to 1).



### 42.3.1.7 Initial Configuration Command Sequence

Before the application starts to screen an update, the SPDC should be configured appropriately.

These rudimentary setup steps are:

1. Send SPDC\_SW\_GATE\_CLK register setting; Disable the clock-gating function.
2. Send SPDC\_INT\_EN register setting; Enable or disable the interrupt function.
3. Send SPDC\_TEMPER\_SETTING register setting; Configure the operated environment temperature.
4. Send SPDC\_NEXT\_BUFF register setting; Configure the next frame memory buffer start address in system memory.
5. Send SPDC\_CURRENT\_BUFF register setting; Configure the current frame memory buffer start address in system memory.
6. Send SPDC\_PREVIOUS\_BUFF register setting; Configure the previous frame memory buffer start address in system memory.
7. Send SPDC\_FRM\_CNT\_BUFF register setting; Configure the counter frame memory buffer start address in system memory.
8. Send SPDC\_LUT\_BUFF register setting; Configure the LUT memory buffer start address in system memory. (If the SPDC\_INT\_EN[LUT\_DOWNLOAD\_FINISH\_INT\_EN] is set to 1, the LUT download finish interrupt will be asserted after this step finish. So that the software designer should clear that interrupt bit field SPDC\_INT\_ST\_CLR[LUT\_DOWNLOAD\_FINISH\_INT\_CLR] after detected the interrupt signal rising.)
9. Send SPDC\_PANEL\_INIT\_SET register setting; Configure the panel resolution, driver scan direction, etc. The most important thing is set the SPDC\_PANEL\_INIT\_SET[POWER\_READY] to 1. (If the SPDC\_INT\_EN[TCON\_INIT\_FINISH\_INT\_EN] is set to 1, the SPDC initialize finish interrupt will be asserted after this step finish. So that the software designer should clear that interrupt bit field SPDC\_INT\_ST\_CLR[TCON\_INIT\_FINISH\_INT\_CLR] after detected the interrupt signal rising.)

### 42.3.2 Display Update Programming

### 42.3.2.1 Basic Waveform Mode Introduction

This section will introduce some typical basic waveform modes for software application development. There are descriptions include mode 0, mode 1, mode 2 and mode 4 given as following:

#### 42.3.2.1.1 Mode 0 - Gray refresh mode with flashing

Features of mode 0:

- 16 level gray (4-bit), foreground is 4 bit and background is 4 bit.
- Require Local update (x,y,w,l) format from Host.
- Black & white Flashing.
- For high quality images.

Figure 45-9 is an example of screen update with mode 0 waveform:



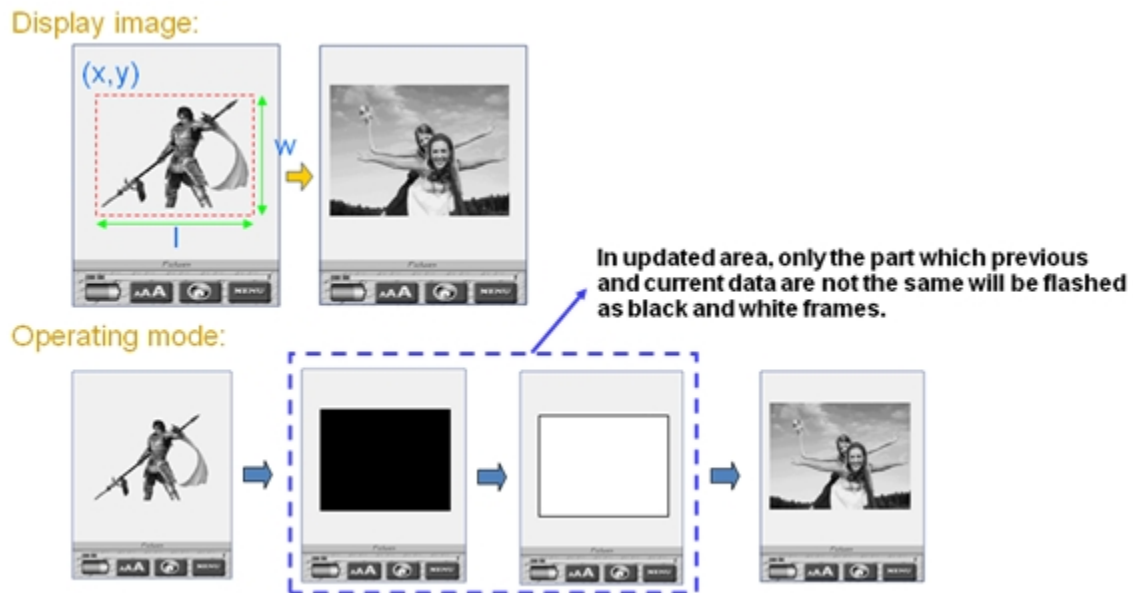
Figure 42-11. The example of screen update with mode 0 waveform

#### 42.3.2.1.2 Mode 1 - Gray refresh mode without flashing

Features of mode 1:

- 16 level gray (4-bit), foreground is 4 bit and background is 4 bit.
- Require Local update (x,y,w,l) format from Host.
- No flashing
- Suit for text/image.

Figure below is an the example of screen update with mode 1 waveform:



**Figure 42-12. The example of screen update with mode 1 waveform**

#### 42.3.2.1.3 Mode 2 - Text mode

Features of mode 2:

- Foreground is 2bit(4 gray level), background is 4 bit
- Require Local update (x,y,w,l) format from Host
- No flashing.
- Suit for text/menu.

Figure below is an the example of screen update with mode 2 waveform:

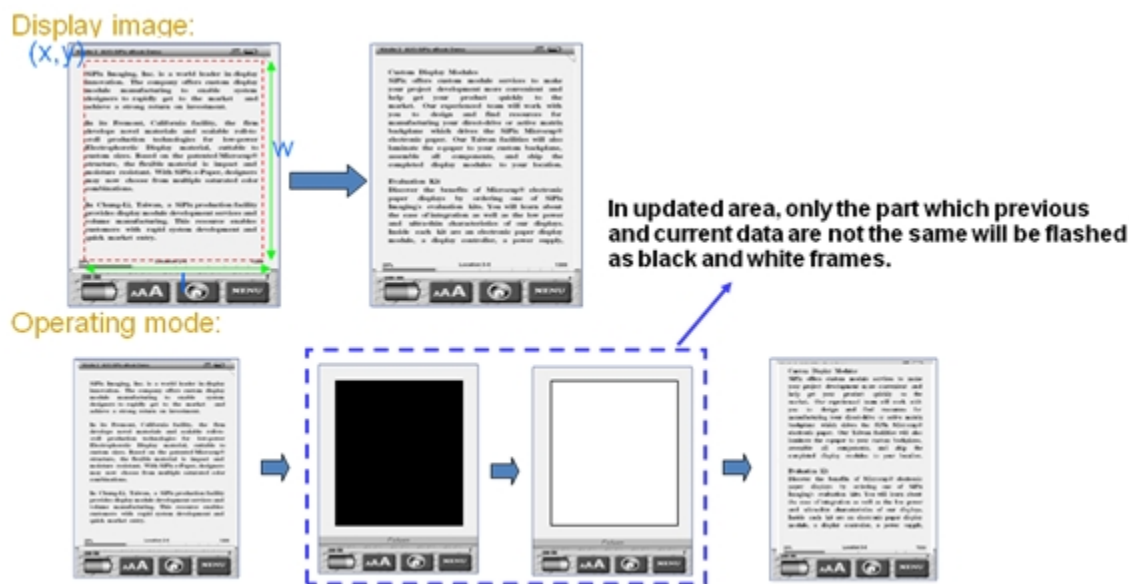


Figure 42-13. The example of screen update with mode 2 waveform

#### 42.3.2.1.4 Mode 4 - High speed handwriting mode

Features of mode 4:

- Handwrite is from white to black, background is 4 bit
- Require Local update (x,y,w,l) format from Host
- No flashing.
- Fast response for handwriting.

Figure below is an the example of screen update with mode 4 waveform:

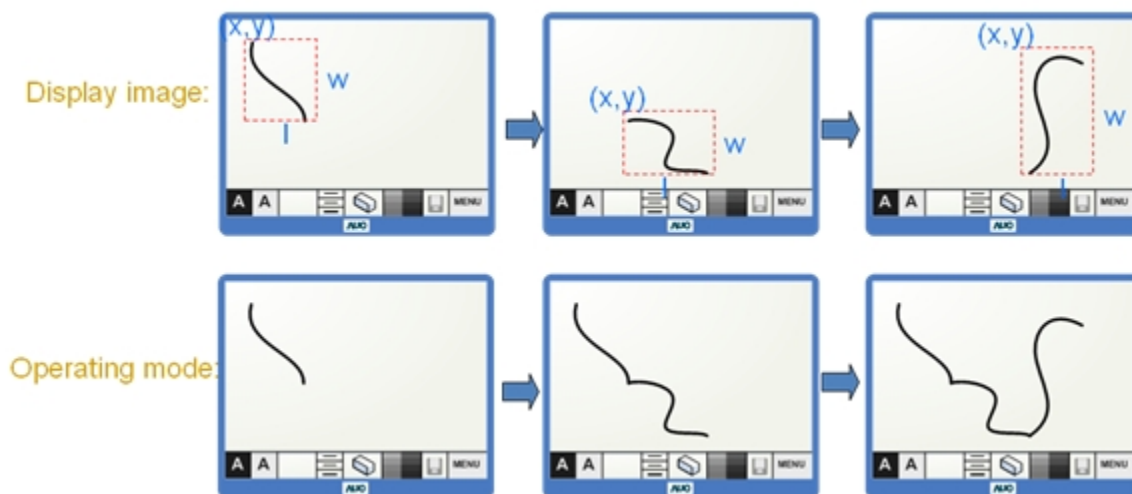


Figure 42-14. The example of screen update with mode 4 waveform

**NOTE**

This section just introduces typical waveform features of each mode but these mode features are maybe changed with different LUT versions.

### **42.3.2.2 Initiating a Display Flow in Normal and High Speed Handwriting Mode**

The typical flow for performing a normal display update mode includes the following steps:

1. Step 1 - Read the SPDC\_STATUS[EPD\_TCON\_BUSY\_N] bit field, it must be 1. The SPDC cannot process new updates if it is currently processing an update.
2. Step 2 - Make sure that the correct temperature is selected and has been written to SPDC\_TEMPER\_SETTING register. The SPDC needs to update the panel screen with the correct environment temperature.
3. Step 3 - The software designer should put the update image into next frame memory buffer in correct location base on the update coordinate (SPDC\_UPDATE\_X\_Y[COORDINATE\_X], SPDC\_UPDATE\_X\_Y[COORDINATE\_Y], SPDC\_UPDATE\_W\_H[WIDTH], SPDC\_UPDATE\_W\_H[HEIGHT]).
4. Step 4 - Send the update coordinate settings such as SPDC\_UPDATE\_X\_Y[COORDINATE\_X], SPDC\_UPDATE\_X\_Y[COORDINATE\_Y], SPDC\_UPDATE\_W\_H[WIDTH], SPDC\_UPDATE\_W\_H[HEIGHT] to SPDC.
5. Step 5 - Send the SPDC\_DISP\_TRIGGER settings to SPDC. It includes screen update trigger and selected waveform mode.

Here we also provide the typical flow for a high speed handwriting mode (mode 4), it is given as following:

1. Step 1 - Make sure that the correct temperature is selected and has been written to SPDC\_TEMPER\_SETTING register. The SPDC needs to update the panel screen with the correct environment temperature.
2. Step 2 - The software designer should put the update image into next frame memory buffer in correct location base on the update coordinate (SPDC\_UPDATE\_X\_Y[COORDINATE\_X], SPDC\_UPDATE\_X\_Y[COORDINATE\_Y], SPDC\_UPDATE\_W\_H[WIDTH], SPDC\_UPDATE\_W\_H[HEIGHT]).
3. Step 3 - Send the update coordinate settings such as SPDC\_UPDATE\_X\_Y[COORDINATE\_X],

SPDC\_UPDATE\_X\_Y[COORDINATE\_Y], SPDC\_UPDATE\_W\_H[WIDTH], SPDC\_UPDATE\_W\_H[HEIGHT] to SPDC.

4. Step 4 - Send the SPDC\_DISP\_TRIGGER settings to SPDC. It includes screen update trigger and selected waveform mode (here should be mode 4).

The difference between a high speed handwriting mode and a normal display update is that the high speed handwriting mode does not need to use the SPDC\_STATUS[EPD\_TCON\_BUSY\_N] status or the update finish interrupt (SPDC\_INT\_ST\_CLR[UPDATE\_FINISH\_INT\_CLR]). This is because in this mode, SPDC will immediately update the panel screen when the software designer sends the screen update trigger (SPDC\_DISP\_TRIGGER[DISP\_TRIG]). It does not need to wait until the SPDC has finished the current screen update (SPDC\_STATUS[EPD\_TCON\_BUSY\_N] set to 1). This makes it much faster than other waveform modes. It is always used in handwriting applications.

### 42.3.2.3 Concurrent Update

The concurrent update in SPDC controller is not a really "concurrent". The controller can accept multiple update command but only can serve one update at a time. Instead of doing the update one by one, the controller will merge all the pending commands into one command. Below are the detailed description of "concurrent update" feature.

1. The controller can accept up to 16 update command;
2. When there is no active command being processed, the first command will start immediately after it is sent to the controller;
3. When there is active command being processed, controller can still accept 15 new update commands, but these command will be pending;
4. When the current active command is done, all the pending commands will be merged into one command automatically by the controller and started automatically;
5. When multiple pending commands are merged, the last command's update mode will be used as the mode for merged command;
6. When multiple pending commands are merged, the update region for merged command will be a rectangle outline just to include the update regions of all the update commands;
7. When a pixel is covered by multiple pending commands' update region, the pixel value in the later update command will be used;

## 42.4 SPDC Memory Map/Register Definition

### SPDC Register Format Summary

**SPDC memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20E_8000	Display Trigger (SPDC_DISP_TRIGGER)	32	R/W	0000_0000h	<a href="#">42.4.1/2668</a>
20E_8004	Display Coordinate (SPDC_UPDATE_X_Y)	32	R/W	0000_0000h	<a href="#">42.4.2/2669</a>
20E_8008	Display Area Size (SPDC_UPDATE_W_H)	32	R/W	0000_0000h	<a href="#">42.4.3/2669</a>
20E_800C	LUT Parameter Update (SPDC_LUT_PARA_UPDATE)	32	R/W	0000_0000h	<a href="#">42.4.4/2670</a>
20E_8010	Display Normal Operation (SPDC_OPERATE)	32	R/W	0000_0000h	<a href="#">42.4.5/2670</a>
20E_8014	SPDC Initial Setting (SPDC_PANEL_INIT_SET)	32	R/W	0000_2000h	<a href="#">42.4.6/2671</a>
20E_8018	Environment Temperature (SPDC_TEMPER_SETTING)	32	R/W	0000_0000h	<a href="#">42.4.7/2674</a>
20E_801C	Next Frame Memory Address (SPDC_NEXT_BUF)	32	R/W	0000_0000h	<a href="#">42.4.8/2675</a>
20E_8020	Current Frame Memory Address (SPDC_CURRENT_BUF)	32	R/W	0000_0000h	<a href="#">42.4.9/2675</a>
20E_8024	Previous Frame Memory Address (SPDC_PREVIOUS_BUFF)	32	R/W	0000_0000h	<a href="#">42.4.10/2676</a>
20E_8028	Counter Frame Memory Address (SPDC_FRM_CNT_BUFF)	32	R/W	0000_0000h	<a href="#">42.4.11/2676</a>
20E_802C	LUT Memory Address (SPDC_LUT_BUFF)	32	R/W	0000_0000h	<a href="#">42.4.12/2676</a>
20E_8030	Interrupt Enable (SPDC_INT_EN)	32	R/W	0000_0000h	<a href="#">42.4.13/2677</a>
20E_8034	Interrupt Status & Clear (SPDC_INT_ST_CLR)	32	R/W	0000_0000h	<a href="#">42.4.14/2678</a>
20E_803C	SPDC Operation Status (SPDC_STATUS)	32	R	0000_0000h	<a href="#">42.4.15/2680</a>
20E_8040	Panel Type Related Information (SPDC_PANEL_TYPE_VER)	32	R	0000_0000h	<a href="#">42.4.16/2681</a>
20E_8044	SPDC IP Version (SPDC_TCON_VER)	32	R	0000_0000h	<a href="#">42.4.17/2682</a>
20E_8048	All Clock Gating Enable (SPDC_SW_GATE_CLK)	32	R/W	0000_0001h	<a href="#">42.4.18/2683</a>

## 42.4.1 Display Trigger (SPDC\_DISP\_TRIGGER)

Address: 20E\_8000h base + 0h offset = 20E\_8000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												FLASH_SET	EPD_MODE		DISP_TRIG
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SPDC\_DISP\_TRIGGER field descriptions

Field	Description
31–5 -	This field is reserved. Reserved
4 FLASH_SET	Judge EPD screen update will be flashed or not in mode 7 0x0 NO_FLASH - It will not be flashed screen update in mode 7 0x1 FLASH - It will be flashed screen update in mode 7
3–1 EPD_MODE	EPD update mode setting 0x0 MODE_0 - It will be displayed with mode 0 waveform 0x1 MODE_1 - It will be displayed with mode 1 waveform 0x2 MODE_2 - It will be displayed with mode 2 waveform 0x3 MODE_3 - It will be displayed with mode 3 waveform 0x4 MODE_4 - It will be displayed with mode 4 waveform (High Speed Handwriting Mode) 0x5 MODE_5 - It will be displayed with mode 5 waveform 0x7 MODE_7 - It will be displayed with mode 7 waveform
0 DISP_TRIG	EPD screen update trigger When set to "1" it will trigger TCON start to display image on panel.



## 42.4.2 Display Coordinate (SPDC\_UPDATE\_X\_Y)

Address: 20E\_8000h base + 4h offset = 20E\_8004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				COORDINATE_X												RSVD1				COORDINATE_Y											
W	Reserved				COORDINATE_X												RSVD1				COORDINATE_Y											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### SPDC\_UPDATE\_X\_Y field descriptions

Field	Description
31–28 -	This field is reserved. Reserved
27–16 COORDINATE_X	Horizontal coordinate of update region (minimum: 1) X should be 4n + 1 format where n is integer
15–12 RSVD1	Reserved
11–0 COORDINATE_Y	Vertical coordinate of update region (minimum: 1) If image have rotated counterclockwise 90 Y should be 4n + 1 format where n is integer

## 42.4.3 Display Area Size (SPDC\_UPDATE\_W\_H)

Address: 20E\_8000h base + 8h offset = 20E\_8008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved				WIDTH												Reserved				HEIGHT											
W	Reserved				WIDTH												Reserved				HEIGHT											
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### SPDC\_UPDATE\_W\_H field descriptions

Field	Description
31–28 -	This field is reserved. Reserved
27–16 WIDTH	Width of update region W should be 4n format where n is integer
15–12 -	This field is reserved. Reserved
11–0 HEIGHT	Height of update region If image have rotated counterclockwise 90 L should be 4n format where n is integer

## 42.4.4 LUT Parameter Update (SPDC\_LUT\_PARA\_UPDATE)

Address: 20E\_8000h base + Ch offset = 20E\_800Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																LUT_PARAMETER_UPDATED_ADDR								LUT_PARAMETER_UPDATED_VALUE							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SPDC\_LUT\_PARA\_UPDATE field descriptions**

Field	Description
31–16 -	This field is reserved. Reserved
15–8 LUT_PARAMETER_UPDATED_ADDR	The address of updated LUT parameter
7–0 LUT_PARAMETER_UPDATED_VALUE	The value of update LUT parameter

## 42.4.5 Display Normal Operation (SPDC\_OPERATE)

Address: 20E\_8000h base + 10h offset = 20E\_8010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SW_TCON_RESET	Reserved														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														DISP_RESET	DEEP_REFRESH
W																DISP_REFRESH
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SPDC\_OPERATE field descriptions**

Field	Description
31 SW_TCON_ RESET	Software TCON register reset When set to "1" it will trigger TCON start to reset all internal registers in TCON.
30–3 -	This field is reserved. Reserved
2 DISP_RESET	Display reset When set to "1" it will trigger TCON start to reset panel screen and automatic update next/current/previous frame memories according to flash LUT setting gray level.
1 DEEP_ REFRESH	Deep refresh (combine display reset and display refresh function) When set to "1" it will trigger TCON start to reset panel screen and automatic refresh panel screen base on DDR image data (current frame memory) with mode 0 waveform.
0 DISP_REFRESH	Display refresh When set to "1" it will trigger TCON start to refresh panel screen base on DDR image data (current frame memory) with mode 0 waveform.

**42.4.6 SPDC Initial Setting (SPDC\_PANEL\_INIT\_SET)****Table 42-12. Resolution Mapping Table of Gray Mode**

Resolution [4:0] (binary)	Resolution Setting (Width x Height)	Portrait / Landscape
11000	Display resolution is 600 x 800.	Portrait
11001	Display resolution is 768 x 1024	Portrait
11010	Reserved	Reserved
11011	Display resolution is 600 x 1024.	Portrait
11100	Display resolution is 825 x 1200	Portrait
11101	Display resolution is 1024 x 1280	Portrait
11110	Display resolution is 1200 x 1600	Portrait
10000	Display resolution is 800 x 1024	Portrait
10001	Display resolution is 825 x 1280	Portrait
10010	Display resolution is 800 x 1280	Portrait
10011	Display resolution is 768 x 1280	Portrait
10100	Display resolution is 960 x 1280	Portrait
00000	Display resolution is 800 x 600.	Landscape
00001	Display resolution is 1024 x 768	Landscape
00010	Reserved	Reserved
00011	Display resolution is 1024 x 600.	Landscape
00100	Display resolution is 1200 x 825	Landscape
00101	Display resolution is 1280 x 1024	Landscape
00110	Display resolution is 1600 x 1200	Landscape

*Table continues on the next page...*

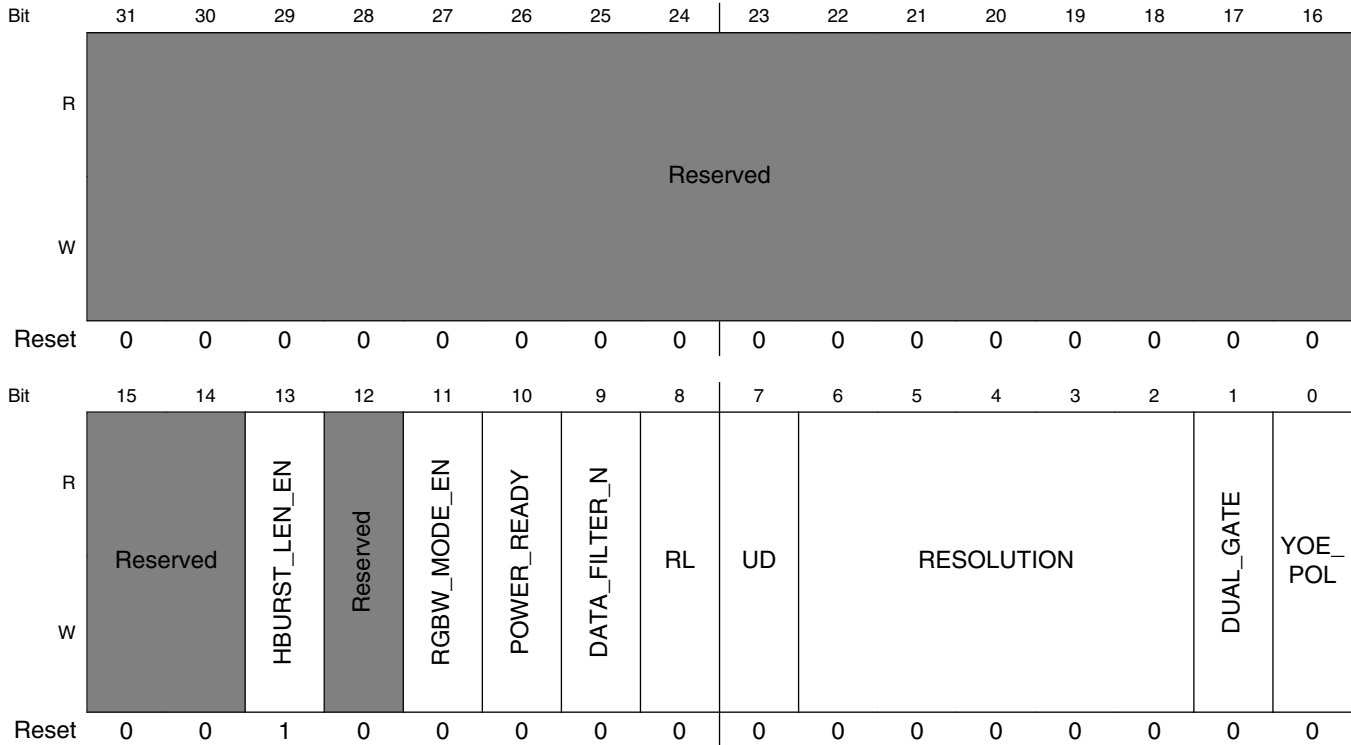
**Table 42-12. Resolution Mapping Table of Gray Mode (continued)**

00111	Display resolution is 1024 x 800	Landscape
01000	Display resolution is 1280 x 825	Landscape
01001	Display resolution is 1280 x 800	Landscape
01010	Display resolution is 1280 x 768	Landscape
01011	Display resolution is 1280 x 960	Landscape
Others	Display resolution is 800 x 600.	Landscape

**Table 42-13. Resolution Mapping Table of RGBW Mode**

Resolution [4:0] (binary)	Resolution Setting (Width x Height)	Portrait / Landscape
11000	Display resolution is 300 x 400.	Portrait
11001	Display resolution is 384 x 512	Portrait
11010	Reserved	Reserved
11011	Display resolution is 300 x 512.	Portrait
11100	Reserved	Reserved
11101	Display resolution is 512 x 640	Portrait
11110	Display resolution is 600 x 800	Portrait
10000	Display resolution is 400 x 512	Portrait
10001	Reserved	Reserved
10010	Display resolution is 400 x 640	Portrait
10011	Display resolution is 384 x 640	Portrait
10100	Display resolution is 480 x 640	Portrait
00000	Display resolution is 400 x 300.	Landscape
00001	Display resolution is 512 x 384	Landscape
00010	Reserved	Reserved
00011	Display resolution is 512 x 300.	Landscape
00100	Reserved	Reserved
00101	Display resolution is 640 x 512	Landscape
00110	Display resolution is 800 x 600	Landscape
00111	Display resolution is 512 x 400	Landscape
01000	Reserved	Reserved
01001	Display resolution is 640 x 400	Landscape
01010	Display resolution is 640 x 384	Landscape
01011	Display resolution is 640 x 480	Landscape
Others	Display resolution is 400 x 300.	Landscape

Address: 20E\_8000h base + 14h offset = 20E\_8014h

**SPDC\_PANEL\_INIT\_SET field descriptions**

Field	Description
31–14 -	This field is reserved. Reserved
13 HBURST_LEN_EN	HBURST length enable 0x0 DISABLE_HBURST - AHB2AXI gasket ignores the HBURST_LEN[3:0]. The AHB INCR will be converted to AXI Singles. 0x1 ENABLE_HBURST -AHB2AXI gasket treats the HBURST_LEN [3:0] as the valid length for current INCR transfer.
12 -	This field is reserved. Reserved
11 RGBW_MODE_EN	RGBW color mode enable 0x0 DISABLE_RGBW_MODE -It will enable TCON into gray mode 0x1 ENABLE_ RGBW_MODE - It will enable TCON into RGBW mode
10 POWER_READY	Driver IC power ready signal 0x0 POWER_NOT_READY - The TCON will be hold wait for driver IC power turn on finish 0x1 POWER_READY - The TCON will into IDLE state and user cloud send the display trigger
9 DATA_FILTER_N	Automatic filter input image pixel data base on the display mode 0x0 ENABLE_AUTO_DATA_FILTER -Filter input data 16-step gray to 4-step gray data for mode 2 and mode 3. Filter input data 16-step gray to 2-step gray for mode 4 and mode 5 0x1 DISABLE_AUTO_DATA_FILTER- It will not be automatic filter the input image data

*Table continues on the next page...*

**SPDC\_PANEL\_INIT\_SET field descriptions (continued)**

Field	Description
8 RL	Select source driver IC scanning direction right or left. 0x0 LEFT -Shift left; First data=Sn ◇ Sn-1 ◇ ...◇ S2 ◇ Last data=S1. 0x1 RIGHT -Shift right: First data=S1◇ S2 ◇ ...◇ Sn-1 ◇ Last data=Sn.
7 UD	Select gate driver IC scanning direction up or down. 0x0 DOWN -Scan down; First line=Gm◇ Gm-1 ◇...◇ G2 ◇ Last line=G1. 0x1 UP -Scan up; First line=G1 ◇ G2 ◇ ... ◇ Gm-1 ◇ Last line=Gm.
6-2 RESOLUTION	Set the panel resolution GRAY_MODE - Please reference the Table 45-12 RGBW_MODE -Please reference the Table 45-13
1 DUAL_GATE	Set to enable the panel type of dual gate driver 0x0 SINGLE_GATE -Single gate driver (support gate driver K7900). 0x1 DOUBLE_GATE -Double side gate driver (support gate driver MEXI2300)
0 YOE_POL	Set the YOE signal's polarity of gate driver IC 0x0 LOW_ENABLE -Gate pulse outputs low enable. 0x1 HIGH_ENABLE -Gate pulse outputs high enable.

**42.4.7 Environment Temperature (SPDC\_TEMPER\_SETTING)****Table 42-15. Temperature Mapping Table**

TEMPERATURE [8:0] (binary)	Thermal Sensor Temperature (oC)
110010010	-55
:	:
111111110	-1
111111111	-0.5
000000000	0
000000001	0.5
000000010	1
:	:
001111111	63.5
:	:

Address: 20E\_8000h base + 18h offset = 20E\_8018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SPDC\_TEMPER\_SETTING field descriptions**

Field	Description
31–9 -	This field is reserved. Reserved
8–0 TEMPERATURE	Update TCON temperature information Temperature Information is used to indicate the temperature of operating environment. TCON IP will search the LUT for a suitable value according to the temperature information. The Table 45-15 is shown the temperature mapping table. (in two's complement format)

**42.4.8 Next Frame Memory Address (SPDC\_NEXT\_BUF)**

Address: 20E\_8000h base + 1Ch offset = 20E\_801Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	<div></div>																															
W	<div></div>																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SPDC\_NEXT\_BUF field descriptions**

Field	Description
31–0 NEXT_FRAME_MEMORY_ADDR	Update next frame memory address of DDR memory The next frame memory address should be word-align value, it means EPD TCON will automatic truncate two bits of LSB.

**42.4.9 Current Frame Memory Address (SPDC\_CURRENT\_BUF)**

Address: 20E\_8000h base + 20h offset = 20E\_8020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CURRENT_FRAME_MEMORY_ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SPDC\_CURRENT\_BUF field descriptions**

Field	Description
31–0 CURRENT_FRAME_MEMORY_ADDR	Update current frame memory address of DDR memory The current frame memory address should be word-align value, it means EPD TCON will automatic truncate two bits of LSB.

## 42.4.10 Previous Frame Memory Address (SPDC\_PREVIOUS\_BUFF)

Address: 20E\_8000h base + 24h offset = 20E\_8024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PREVIOUS_FRAME_MEMORY_ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SPDC\_PREVIOUS\_BUFF field descriptions

Field	Description
31–0 PREVIOUS_FRAME_MEMORY_ADDR	Update previous frame memory address of DDR memory The previous frame memory address should be word-align value, it means EPD TCON wills automatic truncate two bits of LSB.

## 42.4.11 Counter Frame Memory Address (SPDC\_FRM\_CNT\_BUFF)

Address: 20E\_8000h base + 28h offset = 20E\_8028h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	COUNTER_FRAME_MEMORY_ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SPDC\_FRM\_CNT\_BUFF field descriptions

Field	Description
31–0 COUNTER_FRAME_MEMORY_ADDR	Update counter frame memory address of DDR memory The counter frame memory address should be word-align value, it means EPD TCON wills automatic truncate two bits of LSB.

## 42.4.12 LUT Memory Address (SPDC\_LUT\_BUFF)

Address: 20E\_8000h base + 2Ch offset = 20E\_802Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	LUT_MEMORY_ADDR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



**SPDC\_LUT\_BUFF field descriptions**

Field	Description
31–0 LUT_MEMORY_ADDR	Update LUT memory address of DDR memory The LUT memory address should be word-align value, it means EPD TCON wills automatic truncate two bits of LSB.

**42.4.13 Interrupt Enable (SPDC\_INT\_EN)**

Address: 20E\_8000h base + 30h offset = 20E\_8030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												TCON_ERROR_INT_EN	LUT_DOWNLOAD_FINISH_INT_EN	TCON_INIT_FINISH_INT_EN	UPDATE_FINISH_INT_EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SPDC\_INT\_EN field descriptions**

Field	Description
31–4 -	This field is reserved. Reserved
3 TCON_ERROR_INT_EN	Error interrupt signal mask 0x0 DISABLE_ERRPR_INTERRUPT - Disable Error Interrupt 0x1 ENABLE_ERRPR_INTERRUPT - Enable Error Interrupt
2 LUT_DOWNLOAD_FINISH_INT_EN	LUT initial download finish interrupt signal mask 0x0 DISABLE_LUT_DOWNLOAD_FINISH_INTERRUPT - Disable LUT initial Download Finish Interrupt 0x1 ENABLE_LUT_DOWNLOAD_FINISH_INTERRUPT - Enable LUT initial Download Finish Interrupt
1 TCON_INIT_FINISH_INT_EN	TCON initialization finish interrupt signal mask 0x0 DISABLE_TCON_INIT_FINISH_INTERRUPT - Disable TCON Initialization Finish Interrupt

*Table continues on the next page...*

## SPDC\_INT\_EN field descriptions (continued)

Field	Description
	0x1 ENABLE_TCON_INIT_FINISH_INTERRUPT - Enable TCON Initialization Interrupt
0 UPDATE_ FINISH_INT_EN	Screen update finish interrupt signal mask 0x0 DISABLE_UPDATE_FINISH_INTERRUPT - Disable Frame Update Finish Interrupt 0x1 ENABLE_UPDATE_FINISH_INTERRUPT - Enable Frame Update Finish Interrupt

## 42.4.14 Interrupt Status &amp; Clear (SPDC\_INT\_ST\_CLR)

Address: 20E\_8000h base + 34h offset = 20E\_8034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												TCON_ERROR_INT_ CLR	LUT_DOWNLOAD_ FINISH_INT_CLR	TCON_INIT_FINISH_ INT_CLR	UPDATE_FINISH_INT_ CLR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## SPDC\_INT\_ST\_CLR field descriptions

Field	Description
31–4 -	This field is reserved. Reserved
3 TCON_ERROR_ INT_CLR	Error interrupt signal clear When set to "1" it will trigger TCON start to clear error interrupt.
2 LUT_ DOWNLOAD_ FINISH_INT_ CLR	LUT initial download finish interrupt signal clear When set to "1" it will trigger TCON start to clear LUT download finish interrupt.

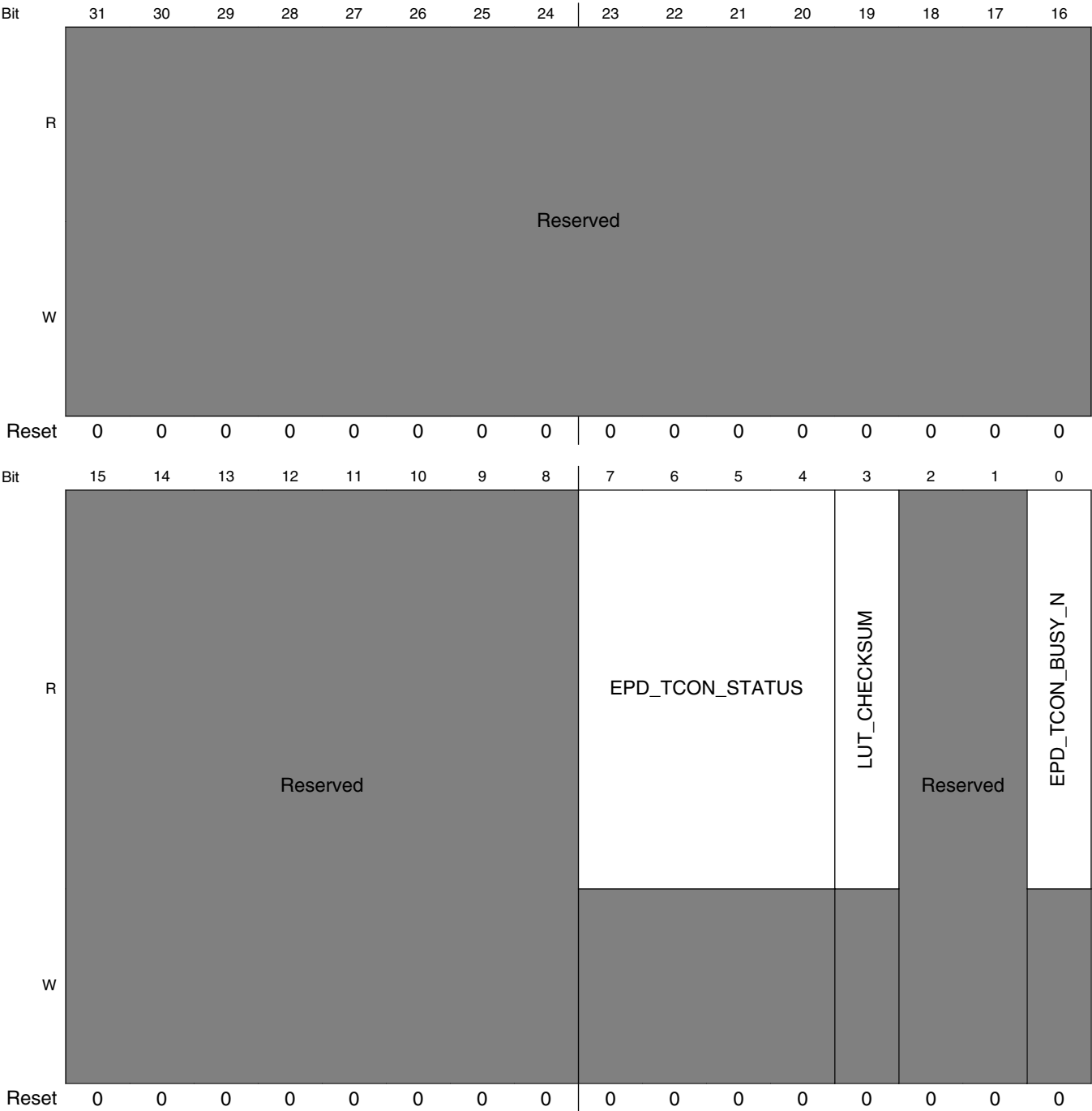
Table continues on the next page...

**SPDC\_INT\_ST\_CLR field descriptions (continued)**

<b>Field</b>	<b>Description</b>
1 TCON_INIT_ FINISH_INT_ CLR	TCON initialization finish interrupt signal clear When set to "1" it will trigger TCON start to clear TCON initialization finish interrupt.
0 UPDATE_ FINISH_INT_ CLR	Screen update finish interrupt signal clear When set to "1" it will trigger TCON start to clear screen update finish interrupt.

### 42.4.15 SPDC Operation Status (SPDC\_STATUS)

Address: 20E\_8000h base + 3Ch offset = 20E\_803Ch



**SPDC\_STATUS field descriptions**

Field	Description
31–8 -	This field is reserved. Reserved
7–4 EPD_TCON_ STATUS	EPD TCON IP status 0x0 RST_STATUS - EPD TCON during reset phase 0x1 WAIT_FOR_CHECKSUM_AND_POWER_READY_STATUS - EPD TCON is hold for waiting checksum pass and driver IC power ready 0x2 CHECKSUM_FAIL_BUT_POWER_READY_STATUS - EPD TCON checksum fail but driver IC power ready. User need reboot EPD TCON again and double check LUT correct or not. 0x3 INITIAL_WORKING_MEMORY_STATUS - EPD TCON automatic initial all working memories 0x4 IDLE_STATUS - EPD TCON idle status 0x7 MOVE_NEXT_TO_CURRENT_MEMORY_STATUS - EPD TCON automatic move next frame memory data to current frame memory. 0x9 LUT_UPDATE_STATUS - EPD TCON automatic update the waveform LUT 0xA PANEL_UPDATE_STATUS - EPD TCON is driving the panel screen.
3 LUT_ CHECKSUM	LUT checksum status 0x0 CHECKSUM_FAIL - LUT checksum fail 0x1 CHECKSUM_SUCCESS - LUT checksum pass
2–1 -	This field is reserved. Reserved
0 EPD_TCON_ BUSY_N	EPD TCON busy status 0x0 BUSY_STATUS - Panel is under updating 0x1 NOT_BUSY_STATUS - Panel update finished

## 42.4.16 Panel Type Related Information (SPDC\_PANEL\_TYPE\_VER)

Address: 20E\_8000h base + 40h offset = 20E\_8040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EPD_PANEL_TYPE								LUT_VERSION								PRODUCT_AND_DRIVER_ID															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SPDC\_PANEL\_TYPE\_VER field descriptions**

Field	Description
31–24 EPD_PANEL_ TYPE	EPD panel type, it stands for different material type of film. 0x0 ERK_1_4_A01 - Material type of film 0x1 ERK_1_4_D01 - Material type of film 0x2 ERK_2_0_A01 - Material type of film

*Table continues on the next page...*

**SPDC\_PANEL\_TYPE\_VER field descriptions (continued)**

Field	Description
	Others need to be defined
23–16 LUT_VERSION	LUT version, it stands for different version of driving waveform 0x0101 V308 - Driving waveform type 0x0102 V312 - Driving waveform type Others need to be defined
15–0 PRODUCT_ AND_DRIVER_ ID	Product ID and driver ID, they are stand for different type of panel 0x0 A060SE02 - Panel type (AUO-K7900) 0x1 A090XE01 - Panel type (AUO-K7900) Others need to be defined

**42.4.17 SPDC IP Version (SPDC\_TCON\_VER)**

Address: 20E\_8000h base + 44h offset = 20E\_8044h

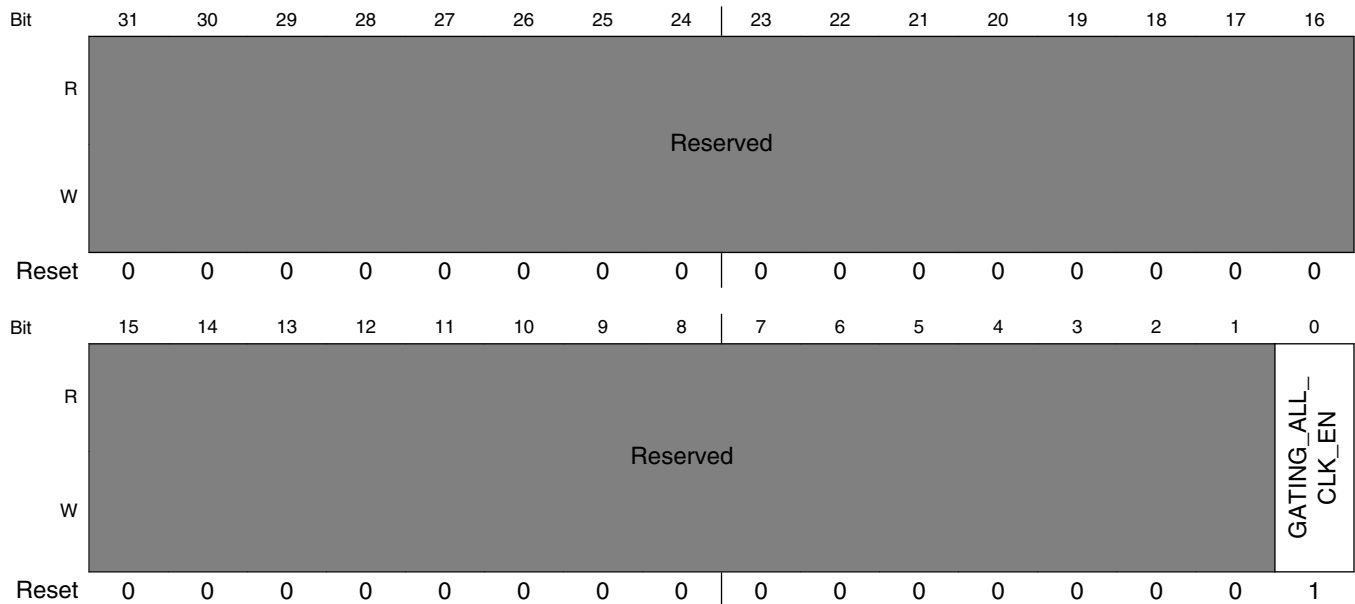
Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																EPD_TCON_VERSION															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SPDC\_TCON\_VER field descriptions**

Field	Description
31–8 -	This field is reserved. Reserved
7–0 EPD_TCON_ VERSION	EPD TCON version, it stands for different version of EPD TCON. 0x0 AUO_T2 - EPD TCON Version Others need to be defined

## 42.4.18 All Clock Gating Enable (SPDC\_SW\_GATE\_CLK)

Address: 20E\_8000h base + 48h offset = 20E\_8048h



**SPDC\_SW\_GATE\_CLK field descriptions**

Field	Description
31–1 -	This field is reserved. Reserved
0 GATING_ALL_CLK_EN	All clocks gating enable 0x0 ENABLE_ALL_CLOCKS - All clocks free run 0x1 GATING_ALL_CLOCKS - Gating all clocks for low power





## **Chapter 43**

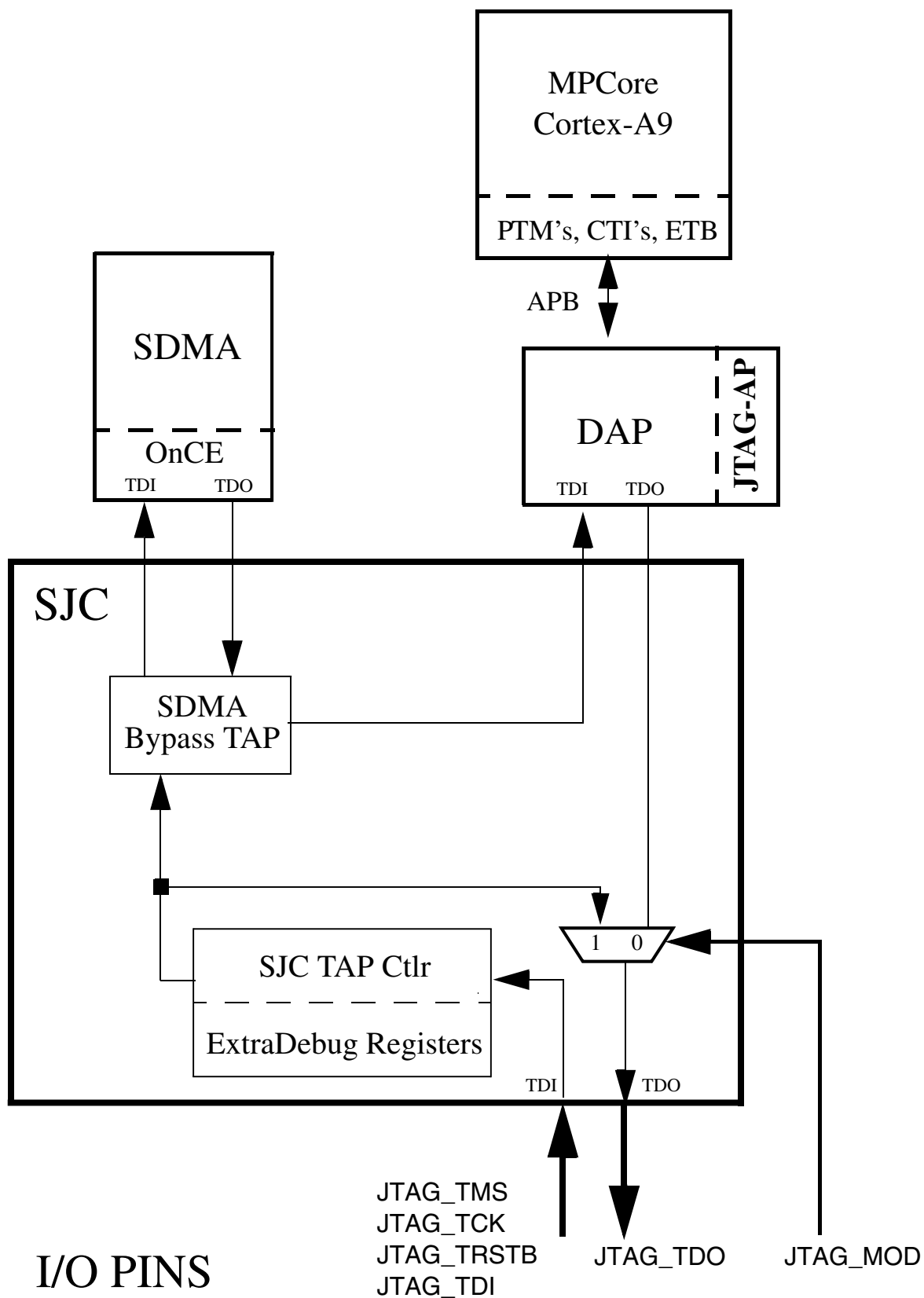
# **System JTAG Controller (SJC)**

### **43.1 Overview**

The System JTAG Controller (SJC) provides debug and test control with the maximum security.

The test access port (TAP) is designed to support features compatible with the IEEE Standard 1149.1 v2001 (JTAG). IEEE P1149.6 standard extensions for AC testing is provided for selected analog IO pads of PCIe and SATA modules.

The figure below shows an overview of the JTAG architecture.



**Figure 43-1. System JTAG Controller (SJC) Block Diagram**  
i.MX 6SoloLite Applications Processor Reference Manual, Rev. 1, 04/2013

### 43.1.1 Features

The System JTAG Controller (SJC) provides the following capabilities:

- JTAG IEEE1149.1 mandatory instructions, see [EXTEST Instruction](#), [SAMPLE/PRELOAD Instruction](#), and [BYPASS Instruction](#).
- JTAG IEEE1149.1 optional instructions, see [ID\\_CODE Instruction \(IDCODE\)](#), and [HIGHZ Instruction](#).
- JTAG IEEE P1149.1 (standard JTAG) interface to off-chip test and development equipment including an SJC-only mode for true IEEE 1149.1 compliance, used primarily for board-level implementation of boundary scan.
- IEEE P1149.6 (JTAG) mandatory instructions, see [EXTEST\\_PULSE instruction](#) and [EXTEST\\_TRAIN instruction](#). These two instructions enable edge-detecting behavior on the signal path containing AC pins.
- Debug-related control and status, such as putting selected cores into reset and/or debug mode and the ability to monitor individual core status signals via JTAG.
- Provides means for accessing each OnCE/ICE TAP controller independently to control a target system (see [Modes of Operation](#)).
- ExtraDebug logic (see [ENABLE\\_ExtraDebug Instruction](#)).
- The maximum clock speed of the SJC is one-eighth of the lowest frequency of the accessed OnCE/ICE. For example in normal operation (no core in low-power mode), this frequency is one-eighth of the SDMA frequency if this core is present in the TDI-TDO chain (serially connected with other cores or standalone). The user must also consider the 25 MHz frequency limitation on the CE bus.
- Core compliant modes to support standalone core debuggers (see [Modes of Operation](#)).
- Multi-cores daisy chained mode (default one) to support multi-core debuggers (see [Modes of Operation](#)).

Detailed information about the SJC is provided in the Security Reference Manual. Contact your Freescale representative for information about obtaining this document.

### 43.1.2 Modes of Operation

The SJC modes are controlled through both the TAP select register (SJC\_TSR) and the MOD input port.

The MOD port (typically connected to pad of the same name) selects between two possible topologies of TAP connections, as seen at SoC level:

- Negating it (this should be the default state) selects all the TAPs ( SJC, SDMA and DAP) to be connected in the TDI-TDO chain, which is referred to as "daisy chain" mode, throughout this chapter.
- Asserting it only selects the SJC TAP to be connected in the TDI-TDO chain.

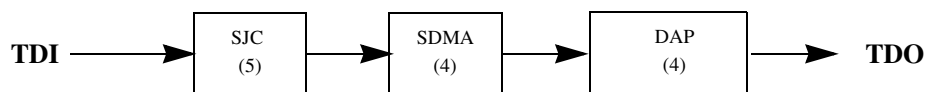
IEEE1149.1 standard features are enabled by configuring the SJC input pin: MOD. Refer to the following table for MOD settings details:

**Table 43-1. SJC Modes**

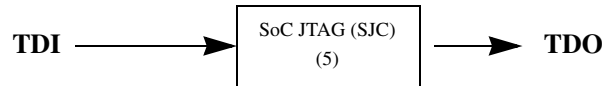
MOD	Name	Description
0	Daisy chain ALL	For common SW debug (High speed and production)
1	SJC only	IEEE 1149.1 JTAG compliant mode

The following figure shows the SJC mode selection flow. The numbers shown in parenthesis below each block name indicates the TAP's IR length.

**MOD = 0**



**MOD = 1**



(number in brackets lists IR length of given TAP)

**Figure 43-2. SJC Mode Selection Using MOD Pin Sampling**

The Connect SDMA bit inside TAP select register controls the SDMA TAP bypass.

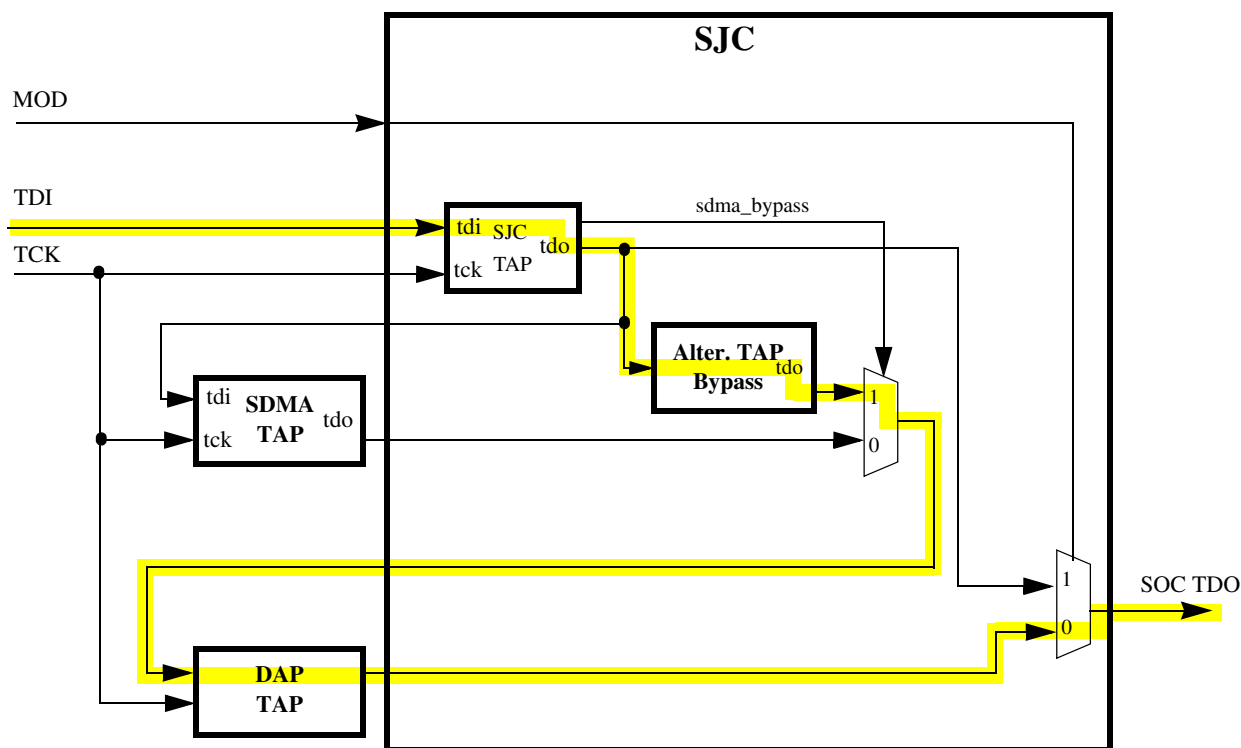
- When negated (should be the default state), the SDMA TAP is bypassed with a single D-FF (Flip-flop) during Shift-Dr path
- When asserted SDMA TAP is connected inside the chain
- When taking the SDMA into bypass or out of bypass (by writing to tapsel reg), additional cycle with TMS '0' should be given

The TAP selection block (TSB) provides a simple method of integrating various pieces of IP that have embedded TAPs.

- Provides a way to connect up multiple TAPs within a single SoC

- Identify the SJC TAP as the master TAP which controls the boundary chain (for IEEE 1149.1 standard compliance)
- Follow the state of SJC TAP, and when the Test-Logic-Reset (TLR) state is reached, reset all TAPs

The figure below shows the TAP Selection Block and SOC TAP Chain Scheme.



Note: The default daisy chain connectivity is highlighted in yellow

**Figure 43-3. TAP Selection Block and SoC TAP Chain Scheme**

### NOTE

It is the responsibility of the user to ensure that in any configuration of the TAP controllers chosen, all of the TAPs in the chain comply with the demands of TCK clock frequency as well as the required ratio between TCK clock frequency and that of the core's to which the TAP refers.

## 43.2 External Signals

The table found here describes the external signals of SJC.

**Table 43-2. SJC External Signals**

Signal	Description	Pad	Mode	Direction
JTAG_DE_B (DE_B)	SoC debug request/acknowledge pin. The DE_IN_B pin is used to propagate an external debug request event to the core(s). This functionality must be enabled first, by set of DE_to_ARM / DE_to_SDMA bits in SJC's DCR register. It is SoC implementation dependent, whether this pin can also be used to reflect the debug acknowledge event back from the cores (in the case where an Open-Drain scheme is used externally).	SD3_DAT1	ALT6	I/O
JTAG_MOD (MOD)	SJC mode selection. This pin is sampled at TRST reset to determine two possible modes for the TAP connection configuration.	JTAG_MOD	No Muxing	I
JTAG_TCK (TCK)	Test Clock (TCK). This is used to synchronize the test logic and includes an internal pull-up resistor	JTAG_TCK	No Muxing	I
JTAG_TDI (TDI)	Test Data Input (TDI). Serial test instruction and data are received through the test data input (TDI) pin. TDI is sampled on the rising edge of TCK and includes an internal pullup resistor	JTAG_TDI	No Muxing	I
JTAG_TDO (TDO)	Test Data Output (TDO). The serial output for test instructions and data. TDO is tri-statable and is actively driven in the shift-IR and shift-DR controller states. TDO changes on the falling edge of TCK	JTAG_TDO	No Muxing	O
JTAG_TMS (TMS)	Test Mode Select (TMS). This is used to sequence the test controller's state machine. TMS is sampled on the rising edge of TCK and includes an internal pullup resistor	JTAG_TMS	No Muxing	I
JTAG_TRSTB (TRSTB)	Test Reset (TRST). This is used to asynchronously initialize the test controller. The TRST pin has an internal pullup resistor	JTAG_TRSTB	No Muxing	I

### 43.2.1 External Signal Overview

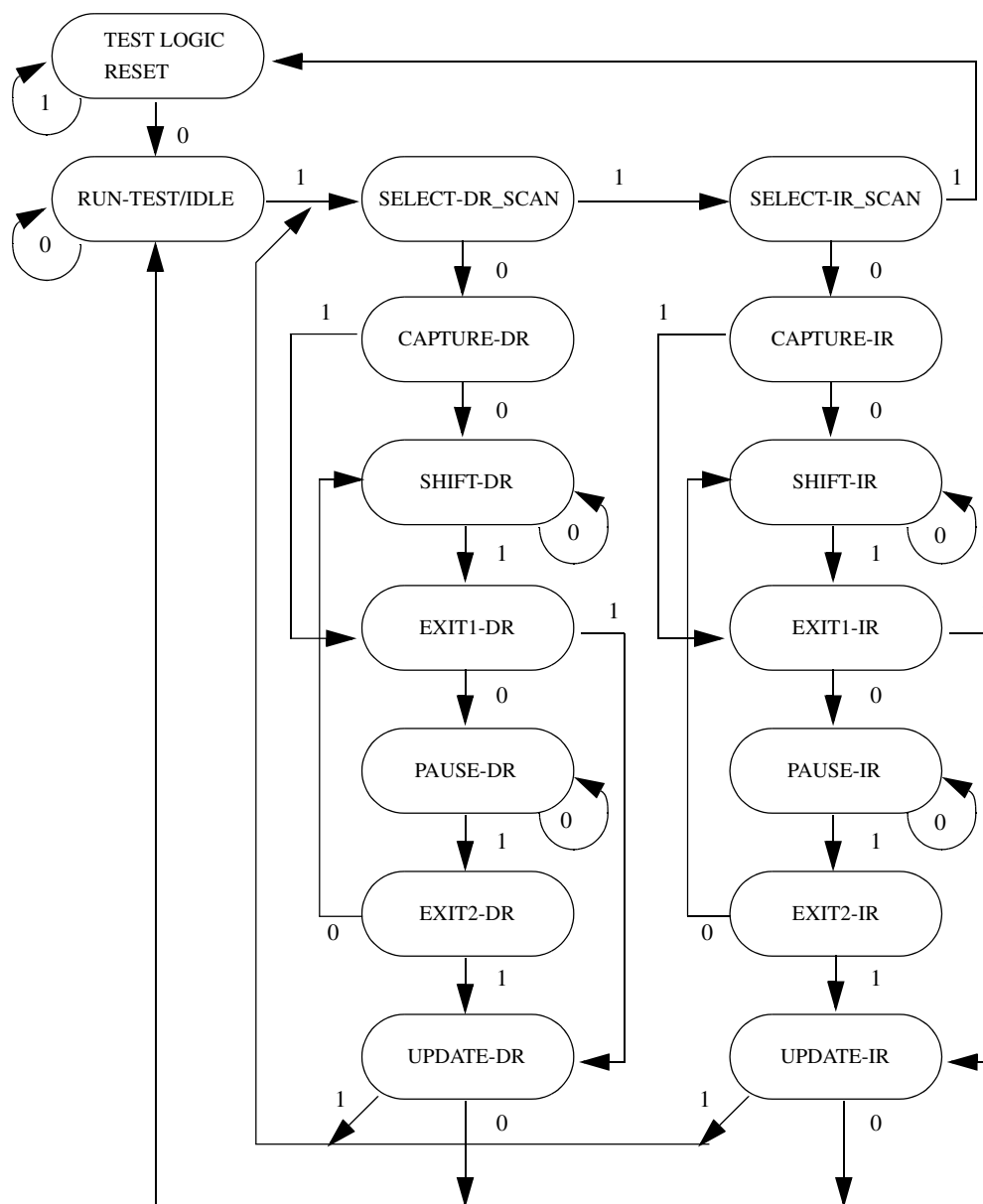
The SJC provides test and debug control with a minimum number of contacts.

The figure below shows SJC connections to external contacts and other chip blocks.

### 43.2.2 TAP Controller

The TAP controller is responsible for interpreting the sequence of logical values on the TMS signal. It is a synchronous state machine that controls the operation of the JTAG logic. The value shown adjacent to each arc represents the value of the TMS signal sampled on the rising edge of TCK signal. For a description of the TAP controller states, refer to the appropriate IEEE 1149.1 document.

The state machine is shown in the following figure.



**Figure 43-4. TAP Controller State Machine**

The change of the JTAG state machine occurs on the rising edge of TCK. TMS and TDI change on the falling edge of TCK. TDO also changes on the falling edge of TCK following entry into the Shift\_DR or Shift\_IR states (TDO\_EN is the enable of the tristate buffer driving the TDO output).

The figure below shows the timings of the SJC signals.



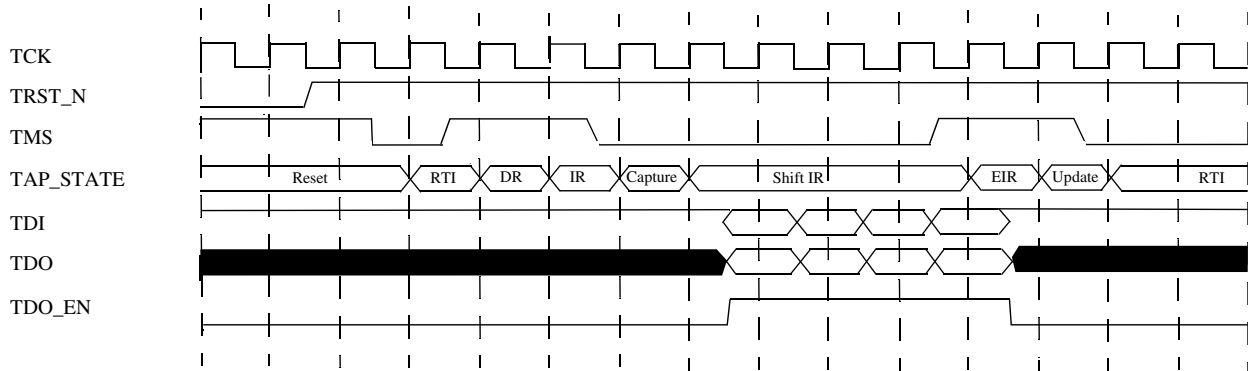


Figure 43-5. SJC Signals Timing Diagram

### 43.2.3 Accessing ExtraDebug Registers

Accessed through the Select-DR-Scan path, the ExtraDebug shift register consists of 38 bits (maximum) comprising a 32-bit data field (max length, see extradebug register description), a 5 bit address field and read/write bit.

The write actually takes place when the JTAG TAP controller enters the Update-DR state. On a read, the data field is ignored (the user should shift only 5 times to enter Read=1 and the address), the read takes place on the next path through DR at the Capture-DR state, the data is shifted-out during the Shift-DR state.

On the second path for a read access, simultaneous write access is not supported: command converter software shifts in zeros so the TAP decodes a write to the CSR (read-only register) which does not have any effect on the circuit.

The number of shift depends on the width of the accessed register as explained in the following diagrams.

First a write access (one path through Select-DR-Scan):

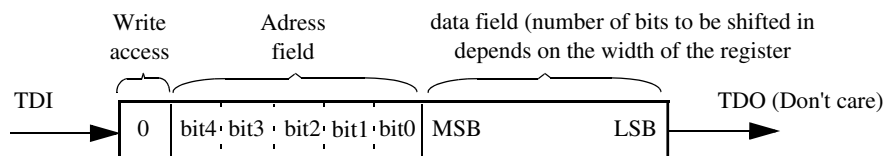
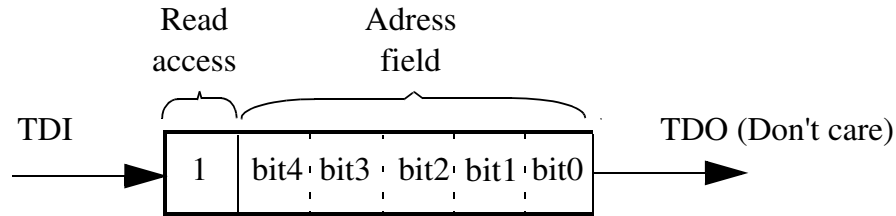


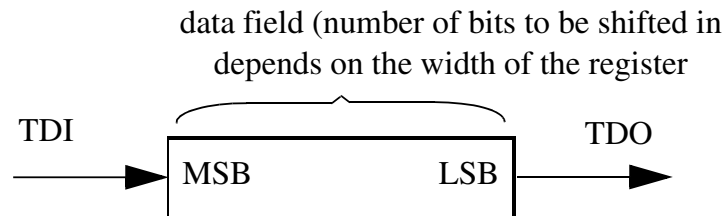
Figure 43-6. TDI/TDO on write access

Then a read access (requires two paths through Jtag DR Scan path):

### *First path*

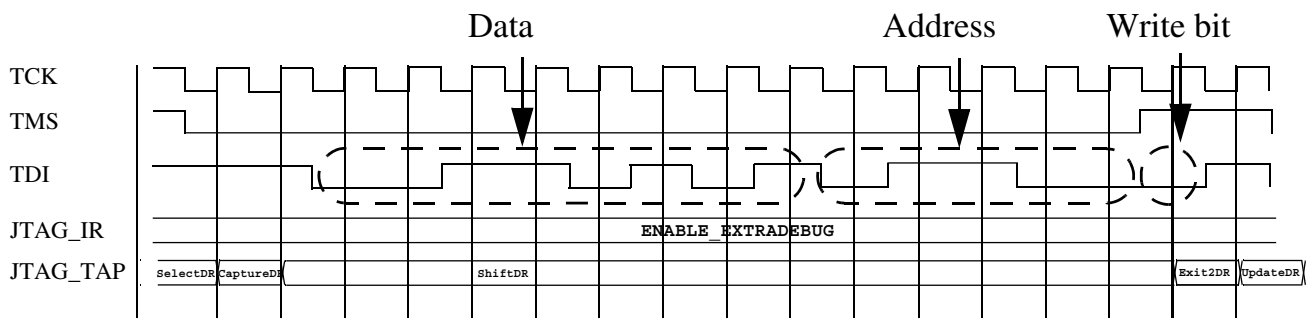


### *Second path*



**Figure 43-7. TDI/TDO on Read Access**

For example, write value 0b1010\_1100 to Debug Control Register (address = 0b00110).



**Figure 43-8. Example: Write Access to DCR**

The SJC registers have different levels of security (refer to [JTAG Security Modes](#)):

- Secured- accessible only in mode 2 (supposed correct response entered), mode 3 and mode 4.
- Unsecured- accessible in all modes

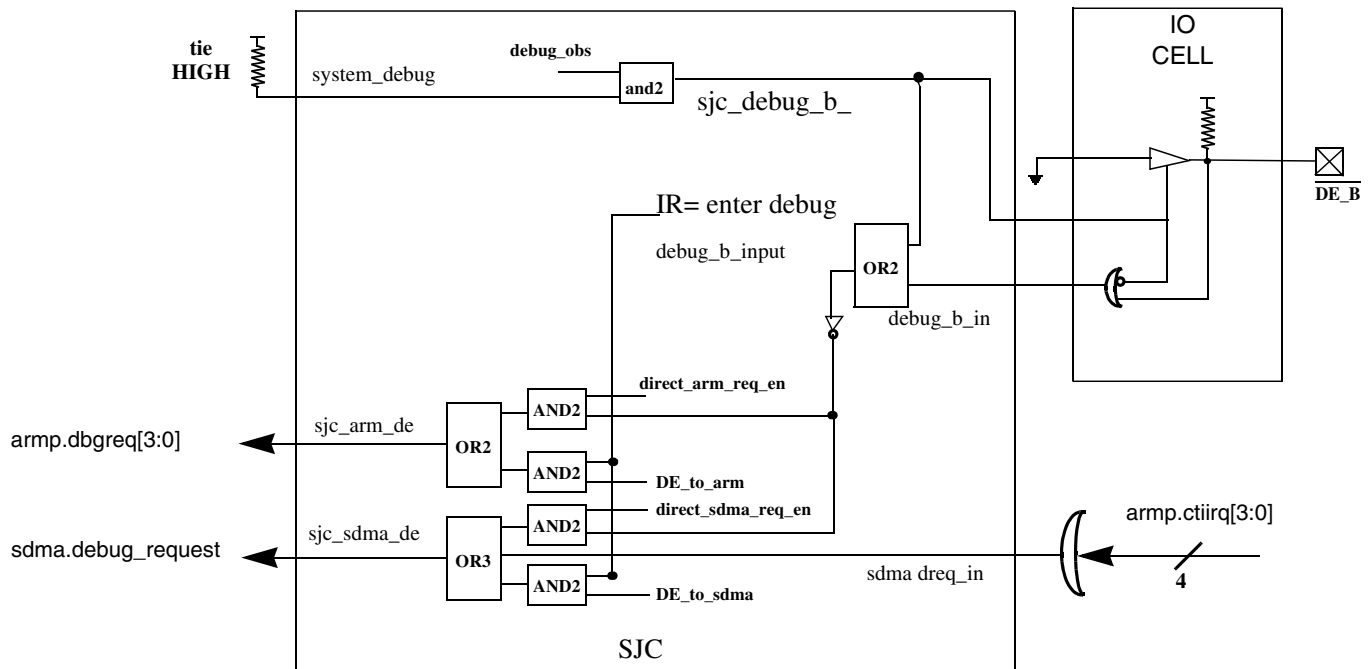
The level of security of each register is indicated in its name or description, in "Programmable Registers" section.

A single DE\_B pin is dedicated for debug request input/output in bidirectional open drain functionality (including an internal pull-up device).

Bits 6:5 in DCR register serve as mask bits, controlling the propagation of external debug request to each recipients (ARM Platform, SDMA).

The bits 1:0 define the propagation enable of IR debug request to recipient cores.

The following figure shows the  $\overline{\text{DE}}$  Pin Select Logic.



**Figure 43-9.  $\overline{\text{DE}}$  Pin Select Logic**

For security reasons, bits for output and input propagation control are at their negated values after reset. A user cannot put the cores in debug mode through  $\overline{\text{DE}}$  without any Jtag access.

The configuration after reset prevents propagation of debug requests / acknowledges to or from the cores.

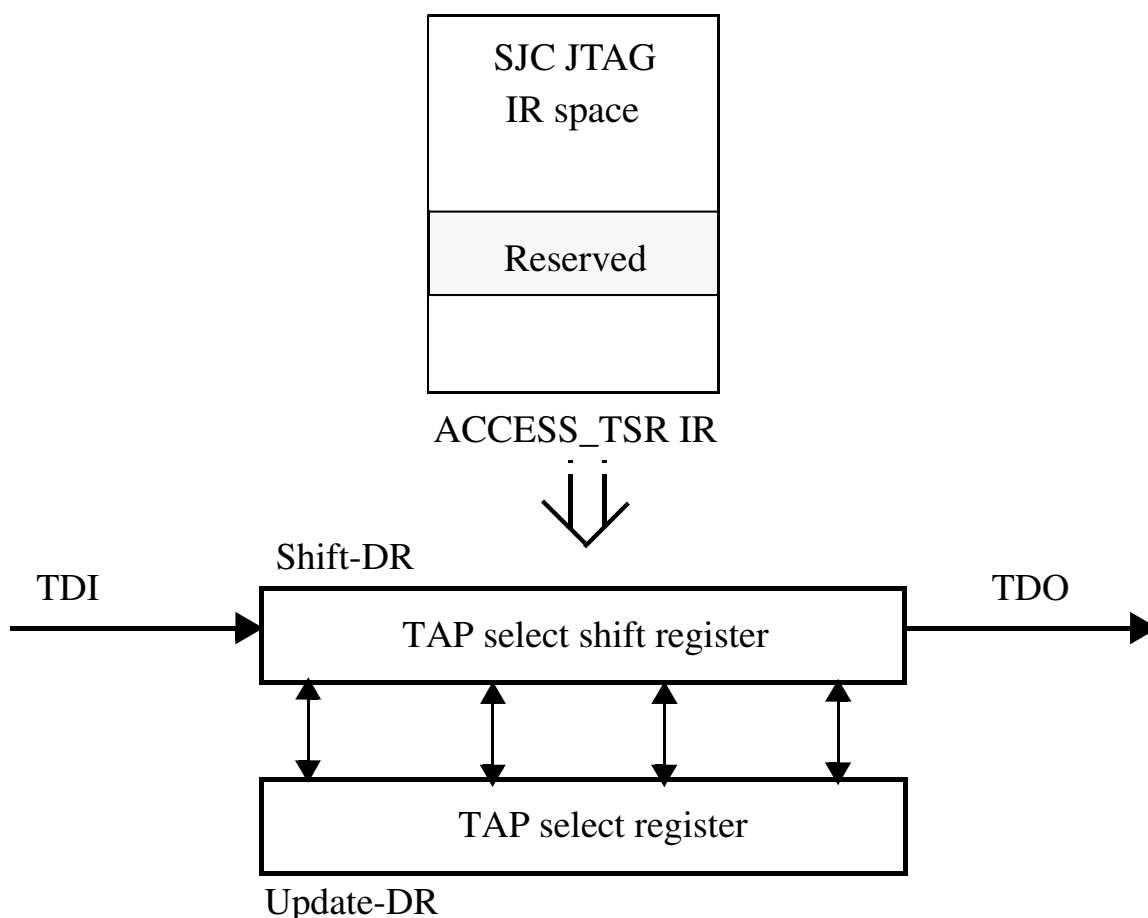
## 43.3 TAP Selection Block (TSB)

As described in [Modes of Operation](#), the SJC can access cores in different modes selected through a TSB.

### 43.3.1 Select Mode Using Software

Conceptually, the SJC\_TSR is a data register which is accessed through Access TSR IR instruction of SJC TAP.

The following figure shows the process of using reserved IR to access the SJC\_TSR.



**Figure 43-10. Using Reserved IR to Access the TAP Select Register (SJC\_TSR)**

The SJC\_TSR can only be changed during the update-DR state of the TSB JTAG state machine. This is necessary to prevent a TAP that is being selected from losing synchronization with the TSB state machine when the TSB state machine returns to run-test-idle. Therefore, an associated shift register for the SJC\_TSR is loaded into the

SJC\_TSR during the update-DR state (see the figure above). The shift register must also capture the state of the SJC\_TSR when in the Capture-DR state for visibility of the contents of the SJC\_TSR. See [TAP Select Instruction](#), for more information.

## 43.4 Boundary Scan Register (BSR)

The Boundary Scan Register (BSR) in the JTAG implementation contains bits for all device signal and clock pins and associated control signals.

All SoC bidirectional pins have a single register bit in the boundary scan register for pin data, and are controlled by an associated control bit in the boundary scan register.

## 43.5 SoC JTAG Instruction Register (SJIR)

The SoC JTAG Instruction register is 5 bits wide.

**Table 43-3. SoC JTAG Instruction Register (SJIR)**

Code					SJC IR
B4	B3	B2	B1	B0	
0	0	0	0	0	IDCODE
0	0	0	0	1	SAMPLE/PRELOAD
0	0	0	1	0	EXTEST
0	0	0	1	1	HI-Z
0	0	1	0	0	ENABLE_ExtraDebug
0	0	1	0	1	ENTER_DEBUG (secured)
0	0	1	1	0	Reserved
0	0	1	1	1	TAP select
0	1	0	0	0	EXTEST_PULSE
0	1	0	0	1	EXTEST_TRAIN
0	1	0	1	0	Reserved
0	1	0	1	1	Reserved
0	1	1	0	0	Security Output challenge
0	1	1	0	1	Security Enter response
-	-	-	-	-	Reserved
1	1	1	1	1	BYPASS

The instruction register is reset to 0b00000 in the test-logic-reset controller state which is equivalent to the IDCODE instruction.

During the capture-IR controller state, the parallel inputs to the instruction register are loaded with the code 01 in the least significant bits as required by the standard; the most significant bits are loaded with the values 00, leading to a capture value of 0b000001.

### 43.5.1 ID\_CODE Instruction (IDCODE)

Selects the ID register, and the system logic controls the I/O pins. This instruction is provided as a public instruction to allow the manufacturer, part number and version of a component to be determined through the TAP.

The table below shows the ID register configuration.

**Table 43-4. ID Configuration Register (IDCODE)**

IDCODE				ID Configuration Register												
	BIT 31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	BIT 16
	Version Information[3:0]				Part Number (Bits 27-16)											
TYPE	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
RESET	0	0	0	0	x	x	x	x	x	x	x	x	x	x	x	x
Note:																
	BIT 15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	BIT 0
	Part Number (Bits 15-12)				Manufacturer Identity											1
TYPE	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r	r
RESET	x	x	x	x	0	0	0	0	0	0	0	1	1	1	0	1
Note:																

**Table 43-5. ID Configuration Register Description (IDCODE)**

Field	Description
31-28 Version Information	IC/SoC Version information number. Initial value: '0000' This number is subject to changes, for new IC/SoC (System On A Chip) revision releases.
27-12 Part Number	Customer Part Number The 16-bit Part Number value is unique for every Freescale's SoC / IC. See "System Debug" chapter for exact register value for a specific SoC.
11-1 Manufacturer Identity	Manufacturer Identity Freescale's Manufacturer Identity code. Bits [11:1] - 00000001110
0	Tied to logic 1.

One application of the ID register is to distinguish the manufacturer(s) of components on a board when multiple sourcing is used. As more components emerge which conform to the IEEE 1149.1 standard, it is desirable to allow for a system diagnostic controller unit to blindly interrogate a board design to determine the type of each component in each location. This information is also available for factory process monitoring and for failure mode analysis of assembled boards.

Once the IDCODE instruction is decoded, it selects the ID register which is a 32 Bit data register. Because the bypass register loads a logic 0 at the start of a scan cycle, whereas the ID register loads a logic 1 into its least significant bit, examination of the first bit of data shifted out of a component during a test data scan sequence immediate following exit from Test-Logic-Reset controller state shows whether such a register is included in the design. When the IDCODE instruction is selected, the operation of the test logic has no effect on the operation of the on-chip system logic as required by the IEEE 1149.1 standard.

### 43.5.2 SAMPLE/PRELOAD Instruction

Selects the boundary scan register and the system logic controls the I/O pins.

The SAMPLE/PRELOAD instruction provides two separate functions:

- First, it provides a means to obtain a snapshot of system data and control signals. The snapshot occurs on the rising edge of TCK in the capture-DR controller state. The data can be observed by shifting it transparently through the boundary scan register.
- The second function of SAMPLE/PRELOAD is to initialize the boundary scan register output cells prior to selection of EXTEST. This initialization ensures that known data appears on the outputs when entering the EXTEST instruction.

#### NOTE

Because there is no internal synchronization between the JTAG clock (TCK) and the system clock (CLK), the user must provide some form of external synchronization to achieve meaningful results.

For more details on the function and use of SAMPLE/PRELOAD, refer to the appropriate IEEE 1149.1 document.

### 43.5.3 EXTEST Instruction

Selects the boundary scan register, and the 1149.1 test logic has control of the I/O pins.

By using the TAP controller, the register is capable of:

- Scanning user-defined values into the output buffers,
- Capturing values presented to input pins
- Controlling the direction of bidirectional pins,
- Controlling the output drive of tri-statable output pins.

For more details on the function and use of EXTEST, refer to the appropriate IEEE 1149.1 document.

The EXTEST instruction also asserts internal reset for the cores (through CCM, refer to [Figure 43-13](#)) to force a predictable internal state while performing external boundary scan operations.

### 43.5.4 HIGHZ Instruction

All output drivers, including the two-state drivers, are turned off (that is, high impedance). The instruction selects the bypass register.

In this mode, all internal pullup resistors on all the pins (except for the TMS, TDI, TCK, TRSTB pins) are disabled. This disabling functionality is not built into SJC, but should be implemented by some logic in the SOC/IO Pads.

For more details on the function and use of HIGHZ, refer to the IEEE 1149.1 document.

The HIGHZ instruction also asserts internal reset for the cores (through CCM, refer to [Figure 43-13](#)) to force a predictable internal state while performing external boundary scan operations.

### 43.5.5 BYPASS Instruction

Selects the single Bit bypass register and the system logic controls the I/O pins.

This creates a shift-register path from TDI to the bypass register and, finally, to TDO, circumventing the boundary scan register. This instruction is used to enhance test efficiency when a component other than the SoC Core based device becomes the device under test.

When the bypass register is selected by the current instruction, the shift-register stage is set to a logic zero on the rising edge of TCK in the capture-DR controller state. Therefore, the first bit to be shifted out after selecting the bypass register is always a logic zero.



For more details on the function and use of BYPASS, refer to the appropriate IEEE 1149.1 document.

### 43.5.6 ENABLE\_ExtraDebug Instruction

The TDI and TDO pins are connected directly to the ExtraDebug registers, the SJC TAP controller remaining connected to TDI and TMS.

The ExtraDebug shift register consists of 38 bits (maximum) comprising a 32-bits data field (maximum length, see [Accessing ExtraDebug Registers](#)), a 5 bits address field and read/write bit. On a register read, the data field does not need to be filled in. The particular ExtraDebug register connected between TDI and TDO at a given time is selected by the ExtraDebug controller depending on the ExtraDebug Address being currently decoded. All communication with the ExtraDebug controller is done through the Select-DR-Scan path of the JTAG TAP Controller.

### 43.5.7 ENTER\_DEBUG instruction

The ENTER\_DEBUG instruction is used to generate a debug request event to SDMA and the ARM MPCore Platform simultaneously (practically, inherited minimal skew is expected, due to difference in event signal propagation in the different modules).

The TDI and TDO are connected to the Instruction Register (IR). After the acknowledgment of the Debug Mode is received (can be checked by reading the Core Status Register part of the ExtraDebug logic), the user can perform system debug functions on the cores.

#### NOTE

The ENTER\_DEBUG event issue to the cores, can be masked, by bits in DCR register.

It is user's responsibility to shift-in another IR value (like IDCODE) before trying to bring the cores out of debug mode, as the debug request signals to the cores remains asserted as long as ENTER\_DEBUG IR is in place.

The user need to check that cores are in debug mode (watching debug acknowledge signal) before leaving ENTER\_DEBUG instruction, otherwise debug request might not take affect.

### 43.5.8 TAP Select Instruction

By means of TAP select instruction a user can access TAP select register and by controlling its only bit SDMA Bypass, control whether SDMA TAP is bypassed or not.

**Table 43-6. TAP Select Register (TSR)**

	TAP Select Register
	BIT 0
	Connect SDMA
TYPE	rw
RESET	0
Note:	

**Table 43-7. TAP Select Register Description**

Field	Description
0 SDMA Bypass	<p>Connect SDMA</p> <p>Control whether SDMA TAP is bypassed or not:</p> <ul style="list-style-type: none"> <li>• 0 - SDMA TAP is bypassed by the alternate TAP inside SJC (emulating 4-bit IR and 1-bit bypass path).</li> <li>• 1 - SDMA TAP is connected to the TDI-TDO chain.</li> </ul> <p><b>NOTE:</b> Additional cycle with TMS '0' should be inserted, after writing to this register, to allow the SDMA tap be sync before SDMA get into / out of bypass.</p>

### 43.5.9 EXTEST\_PULSE instruction

The EXTEST\_PULSE instruction implements new test behaviors for AC pins and simultaneously behaves identically to IEEE Std 1149.1 EXTEST for DC pins.

### 43.5.10 EXTEST\_TRAIN instruction

The EXTEST\_TRAIN instruction implements new test behaviors for AC pins and simultaneously behaves identically to IEEE Std 1149.1 EXTEST for DC pins.

## 43.6 Security

JTAG manipulation is one of the known hackers' ways of executing unauthorized program code, getting control over the OS and run code in privileged modes.

The SJC provides a debug access to several H/W blocks including the ARM processor and the system bus. This allows for program control and manipulation as well as visibility into system peripherals and memory. The PTM interface allow bus transactions to be traced. Together these tools provide the hacker all the access needed to completely comprise the system. Means must be provided to block any malicious JTAG access.

The SJC provides a way of regulating the JTAG access.

The following are the different JTAG security modes:

- Mode #1: No Debug-Maximum Security. All security sensitive JTAG features are permanently blocked.
- Mode #2: Secure JTAG-High security. JTAG use is regulated by secret key based authentication mechanism.
- Mode #3: JTAG Enabled-Low security. JTAG always enabled.

The JTAG security modes are configured using eFUSES which can be burned after packaging by applying electrical signals. The fuse burning is an irreversible process, once a fuse is burned (e-fuse or laser fuse) it is impossible to change the fuse back to the un-burned state.

## 43.6.1 JTAG Security Modes

JTAG can be in one of JTAG security modes which is selected by setting the SJC eFUSE configuration. The physical location of the fuses is not in the SJC.

### 43.6.1.1 Mode 1: No Debug - Maximum Security

No Debug JTAG security mode provides the highest security level.

In this mode, all JTAG features are disabled except for:

- ScanBoundary Scan
- MBIST, all modes except for debug modes which enable controlled memory contents output
- PLL BIST
- BIST monitor mode, allowing routing to external pins BIST pass/fail/invoke information
- PLL bypass- Bypass ARM or/and USB PLL.
- Visibility of the following status bits: power mode - normal, standby, stop, shutdown, and so on

These features do not reduce the security level of the product, and they allows to perform important tests and board connectivity checks.

### **43.6.1.2 Mode 2: Secure JTAG - High Security**

The Secure JTAG mode limits the JTAG access by using challenge/response based authentication mechanism. Any access to JTAG port is being checked. Only authorized debug devices (that is, devices having the right response) can access the JTAG, unauthorized JTAG access attempts are denied.

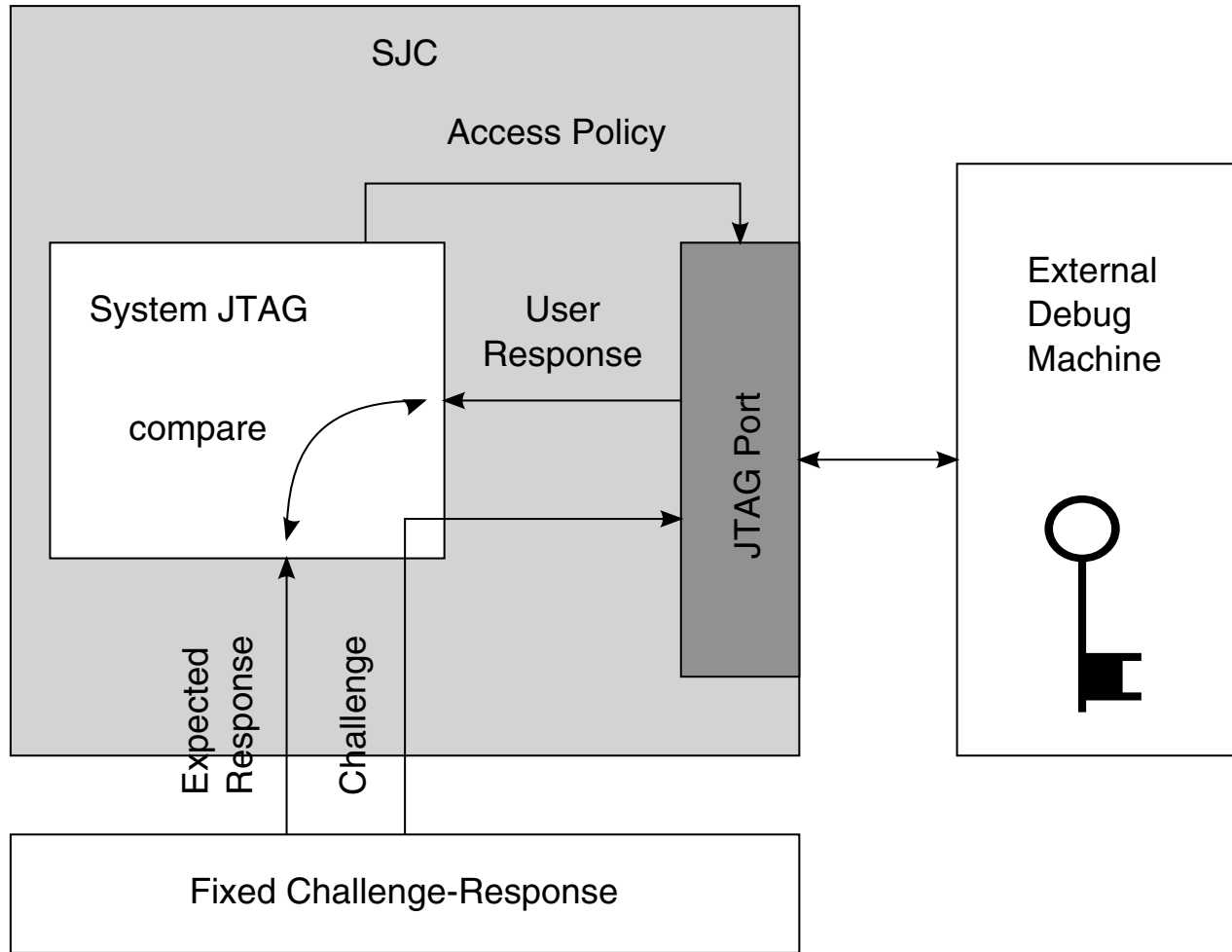
The intent of this mode is to allow return field testing. When a secured JTAG device is being returned for debugging, this mode allows authorized re-activation of the JTAG.

#### **43.6.1.2.1 Challenge/Response Mechanism in System JTAG Mode**

When SJC is in Sysytem JTAG mode the authentication process is as follows:

1. Shift Output Challenge instruction to IR.
2. Passing through Capture-DR state of the SJC and by performing Shift-DR operations Challenge code can be accessed from TDO.
3. Shift Enter Response instruction to IR. By performing Shift-DR, operations enter Response code value through TDI. As Update-DR state is entered, Response code is compared with the correct one.

In Fixed challenge-response pair mode, each part has its individual challenge - response pair which is determined at manufacturing time, and does not change later on. The SJC compares the user's response to the expected response.



**Figure 43-11. Mode #2 - Secure JTAG with Fixed Challenge-response Pair**

### 43.6.1.3 Mode 3: JTAG Enabled - Low Security

In the JTAG Enabled JTAG security mode, all JTAG features are enabled.

## 43.6.2 Software Enabled JTAG

To increase the flexibility of the SJC, an option to enable the JTAG via software is added and is available only in Secure JTAG mode. By writing '1' to HAB\_JDE (HAB JTAG DEBUG ENABLE) bit in the e-fuse controller module ( OCOTP\_CTRL), the JTAG is opened, regardless of its security mode. It is the responsibility of software to assert or negate this bit.

Additionally, a corresponding lock bit is available (in the e-fuse control module) to ensure that only trusted software is able to set the JDE bit. When the LOCK bit is set, no future change of JDE is possible, until the next POR (power-on-reset) cycle.

The platform initialization software should set the LOCK bit for JDE bit before transferring control to the application code.

The S/W JTAG enable allows JTAG enabling without activating the challenge-Response mechanism (which requires JTAG access tool enhancement or special H/W). The JTAG S/W enable does not allow debug in case of boot or memory fault as it requires reset before entering debug.

This feature can be permanently blocked by burning the dedicated e-fuse.

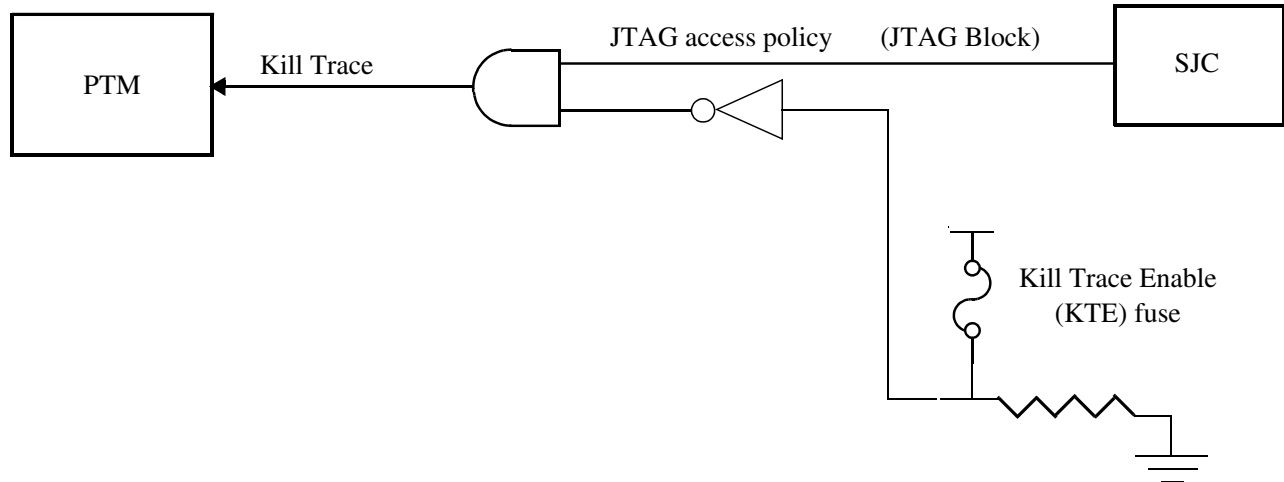
### **NOTE**

The S/W enabled JTAG feature reduces the overall security level of the system as it relies on S/W protections. If this feature is not required, it is strongly recommended to burn the JTAG\_HEO e-fuse which disables this feature.

## **43.6.3 Kill Trace**

The kill trace signal disables any output of the PTM block. The PTM can be accessed either via JTAG port and/or by direct software code. Blocking the JTAG port also yields assertion of the kill trace signal. This resulted in blocking of trace port. The intention of this action is to block any attempt to break into the system via software manipulation of the debug modules. The kill trace, when active, prevents trace output even in case where it can be activated via chip pin.

The kill trace feature needs to be activated by burning a dedicated e-fuse. If the fuse is left intact, kill trace is never activated as seen in [Figure 43-12](#).



**Figure 43-12. Kill Trace eFUSE**

The kill trace is asserted when "kill trace enable" fuse is burned and "ipt\_secur\_block" signal in SJC is asserted, which happens when at least one of the following is true:

- Mode #2 (Secure JTAG) and no code has been entered
- Mode #2 (Secure JTAG) with incorrect response entered
- Mode #1 (No debug)
- TRST\_B signal is active
- POR has not ever been asserted

#### 43.6.4 SJC Disable Fuse

In addition to the different JTAG security modes that are implemented internally in the System JTAG Controller (SJC), there is an option to disable the SJC functionality by eFUSE configuration. This creates additional JTAG mode that is, JTAG Disabled with highest level of JTAG protection. In this mode all JTAG features are disabled.

Specifically, the following debug features are disabled in addition to the features that were already disabled in No Debug JTAG mode:

- Memory BIST
- Boundary scan register (SJC\_BSR)
- Non-Secure JTAG control registers (PLL configuration, Deterministic Reset, PLL bypass)
- Non-Secure JTAG status registers (Core status)
- Chip Identification Code (IDCODE)

## 43.7 Functional Description

This section provides a complete functional description of the block.

### 43.7.1 Static Core Debug

The SJC JTAG TAP controller is fully compatible with the IEEE 1149.1a-2001 Standard Test Access Port and Boundary Scan Architecture specifications.

The ARM MPCore platform debug system (named CoreSight) including the real-time Program Trace Macrocell (PTM), are controlled via the Debug Access Port (DAP) module. Refer to ARM MPCore and PTM Technical reference manuals for more details.

The SDMA has a TAP controller to manage its own OnCE, see SDMA OnCE specifications for more details.

The OnCE and ICE provide a mean of interacting with the cores and their peripherals non-intrusively so that a user may examine registers, memories to facilitate hardware and software development. Refer to [TAP Selection Block \(TSB\)](#), for more information.

### 43.7.2 Reset Mechanism

The following figure shows the SJC reset logic



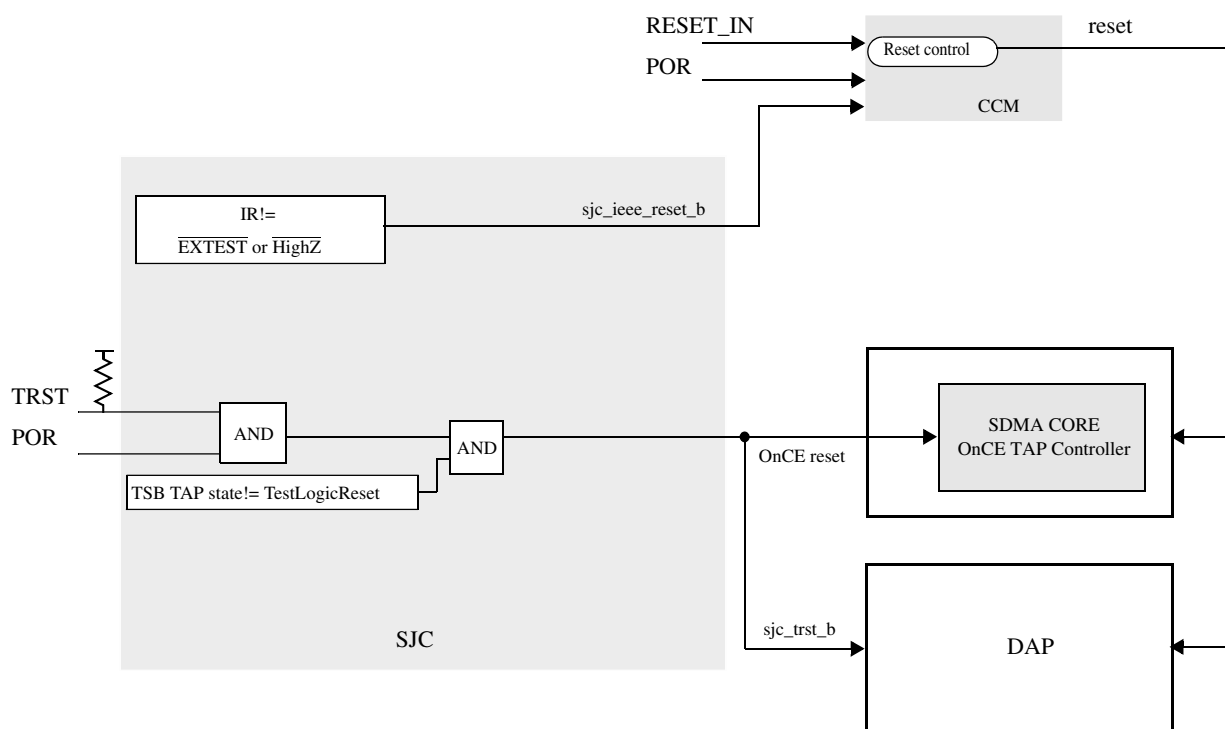


Figure 43-13. SJC Reset Logic

**NOTE**

- Asserting TRSTB in any scan mode resets the TCR losing the testmode configuration and selects default TAP.
- SJC generates an IEEE reset signal to the CCM when in one of the IEEE modes HIGHZ or EXTEST. This signal generates a system reset to the cores until exit from one of these modes.
- The TSB generates Once/ICE reset (either TRSTB if implemented or other) when its TAP state reaches Test-Logic-Reset (meaning that TAP accessed is also reaching Test-Logic-Reset).

## 43.8 Initialization/Application Information

The control afforded by the output enable signals using the boundary scan register and the EXTEST instruction requires a compatible circuit-board test environment to avoid device-destructive configurations. The user must avoid situations in which the SJC output drivers are enabled into actively driven networks.

There are two constraints related to the JTAG interface:

- Ensure that the JTAG test logic is kept transparent to the system logic by forcing TAP into the Test-Logic-Reset controller state. During power-up, SJC's internal TRSTB is asserted as IC's POR\_B is asserted which forces the TAP controller into this state. After that, if TMS either remains unconnected or is connected to VCC, then the TAP controller cannot leave the Test-Logic-Reset state, regardless of the state of TCK.
- DE\_B is an IO pin with pullup and care must be taken of the direction when driving this signal.

## 43.9 SJC Memory Map/Register Definition

In addition to the standard accessible JTAG registers (per IEEE1149.1 standard) listed in [SoC JTAG Instruction Register \(SJIR\)](#) , the chip contains the following registers accessed using the ExtraDebug mechanism, controlled via "ENABLE\_ExtraDebug" IR instruction.

### NOTE

SJC registers are only accessible by JTAG interface. They are not memory mapped to processor address space, so the absolute addresses provided by default in the SJC memory map are not valid.

This section assumes the JTAG controller is accessed in standalone mode or daisy chained (defined by TAP Selection Block) using the appropriate TSB configuration.

See "System Debug" chapter for more details about the general purpose register descriptions that are unique to this chip.

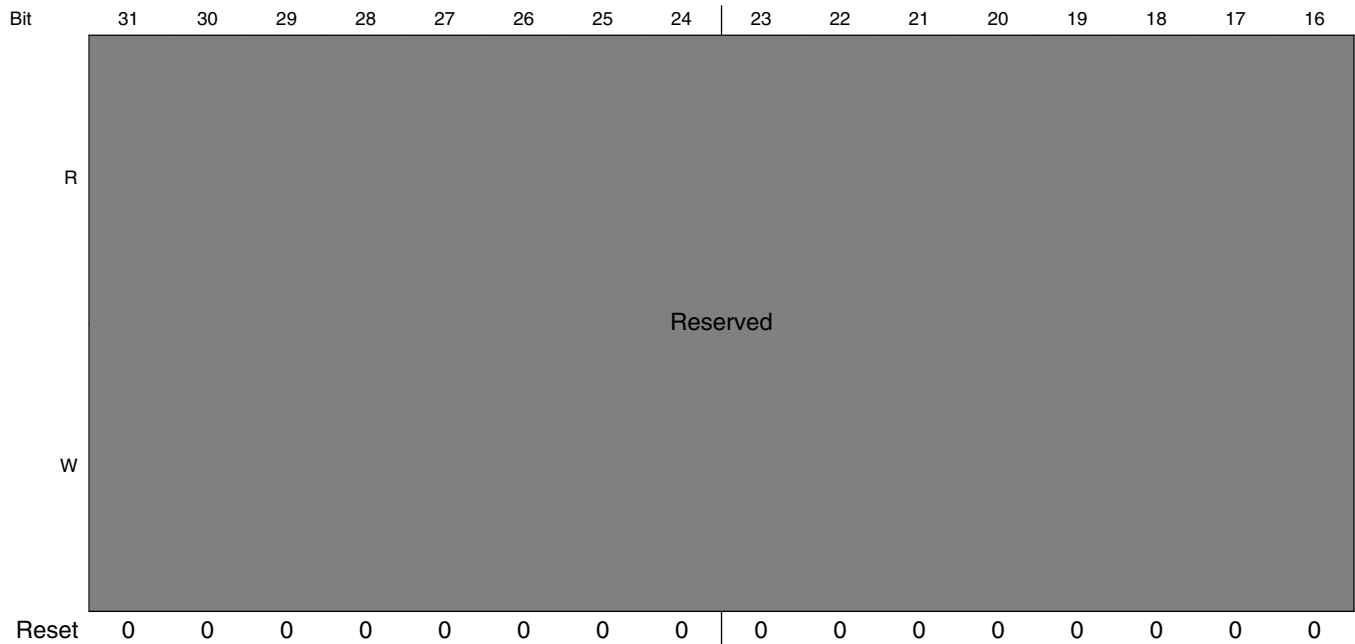
**SJC memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
0	General Purpose Unsecured Status Register 1 (SJC_GPUSR1)	32	R	0000_0000h	<a href="#">43.9.1/2711</a>
1	General Purpose Unsecured Status Register 2 (SJC_GPUSR2)	32	R	0000_0000h	<a href="#">43.9.2/2713</a>
2	General Purpose Unsecured Status Register 3 (SJC_GPUSR3)	32	R	0000_0000h	<a href="#">43.9.3/2713</a>
3	General Purpose Secured Status Register (SJC_GPSSR)	32	R	0000_0000h	<a href="#">43.9.4/2714</a>
4	Debug Control Register (SJC_DCR)	32	R/W	0000_0000h	<a href="#">43.9.5/2715</a>
5	Security Status Register (SJC_SSR)	32	R	<a href="#">See section</a>	<a href="#">43.9.6/2717</a>
7	General Purpose Clocks Control Register (SJC_GPCCR)	32	R/W	0000_0000h	<a href="#">43.9.7/2720</a>

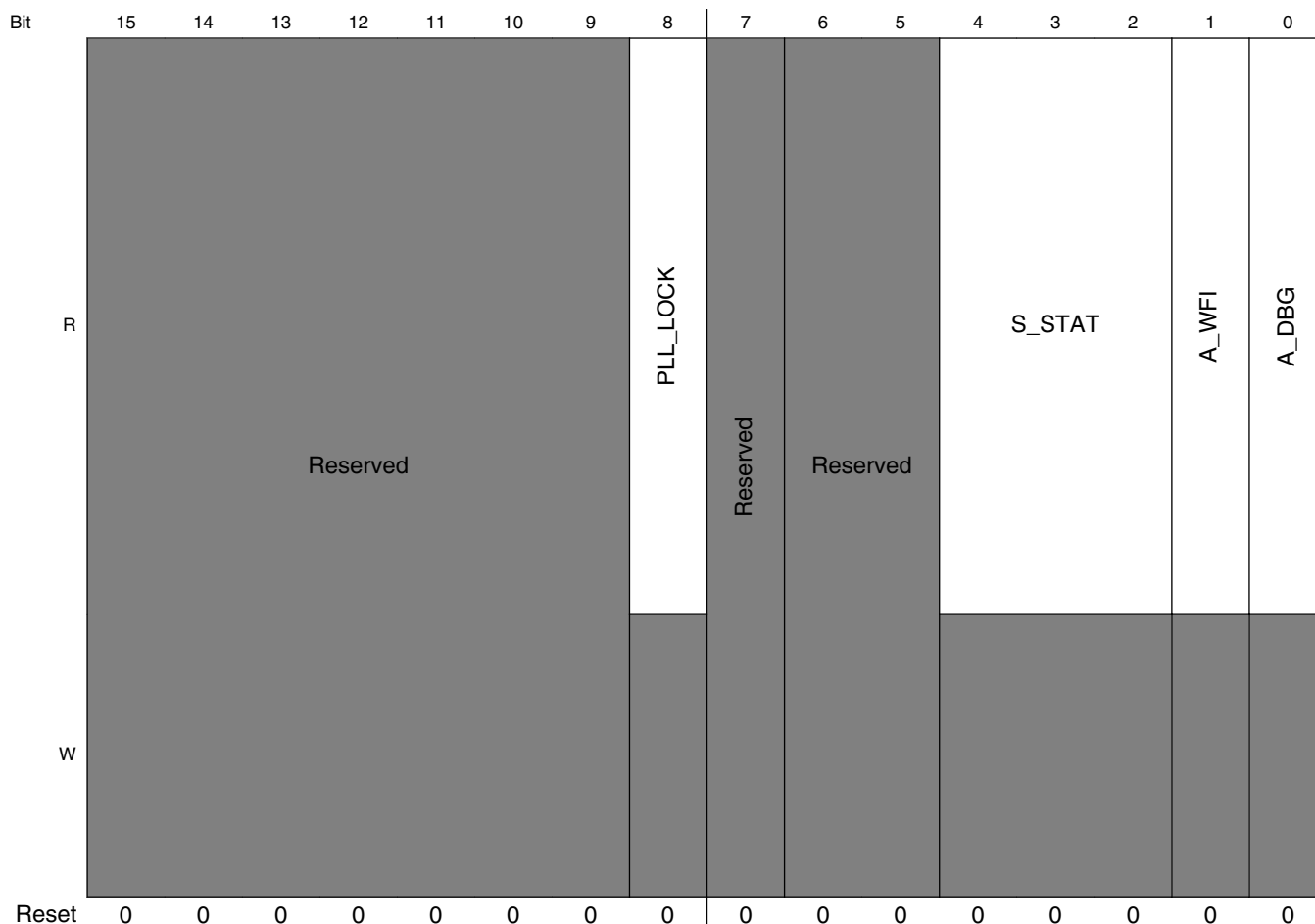
### 43.9.1 General Purpose Unsecured Status Register 1 (SJC\_GPUSR1)

The General Purpose Unsecured Status Register 1 is a read only registers used to check the status of the different Cores and of the PLL. The rest of its bits are for general purpose use.

Address: 0h base + 0h offset = 0h



## SJC Memory Map/Register Definition



### SJC\_GPUSR1 field descriptions

Field	Description
31–9 -	This field is reserved. Reserved.
8 PLL_LOCK	PLL_LOCK A Combined PLL-Lock flag indicator, for all the PLL's.
7 -	This field is reserved. Reserved
6–5 -	This field is reserved. Reserved.
4–2 S_STAT	3 LSBits of SDMA core statusH.
1 A_WFI	ARM core wait-for interrupt bit Bit 1 is the ARM core standbywfi (stand by wait-for interrupt). When this bit is HIGH, ARM core is in wait for interrupt mode.
0 A_DBG	ARM core debug status bit Bit 0 is the ARM core DBGACK (debug acknowledge) DBGACK can be overwritten in the ARM core DCR to force a particular DBGACK value. Consequently interpretation of the DBGACK value is highly dependent on the debug sequence. When this bit is HIGH, ARM core is in debug.

## 43.9.2 General Purpose Unsecured Status Register 2 (SJC\_GPUSR2)

Address: 0h base + 1h offset = 1h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																STBYWFE				S_STAT				STBYWFI							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SJC\_GPUSR2 field descriptions

Field	Description
31–12 -	This field is reserved. Reserved
11–8 STBYWFE	STBYWFE[3:0] Reflecting the "Standby Wait For Event" signals of all cores.
7–4 S_STAT	S_STAT[3:0] SDMA debug status bits: debug_core_state[3:0]
3–0 STBYWFI	STBYWFI[3:0] These bits provide status of "Standby Wait-For-Interrupt" state of all ARM cores.

## 43.9.3 General Purpose Unsecured Status Register 3 (SJC\_GPUSR3)

Address: 0h base + 2h offset = 2h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												SYS_WAIT	IPG_STOP	IPG_WAIT	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SJC\_GPUSR3 field descriptions

Field	Description
31–3 -	This field is reserved. Reserved

Table continues on the next page...

**SJC\_GPUSR3 field descriptions (continued)**

Field	Description
2 SYS_WAIT	System In wait Indication on System in wait mode (from CCM).
1 IPG_STOP	IPG_STOP CCM's "ipg_stop" signal indication
0 IPG_WAIT	IPG_WAIT CCM's "ipg_wait" signal indication

**43.9.4 General Purpose Secured Status Register (SJC\_GPSSR)**

The General Purpose Secured Status Register is a read-only register used to check the status of the different critical information in the SoC. This register cannot be accessed in secure modes.

Address: 0h base + 3h offset = 3h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPSSR																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SJC\_GPSSR field descriptions**

Field	Description
31–0 GPSSR	General Purpose Secured Status Register Register is used for testing and debug.

### 43.9.5 Debug Control Register (SJC\_DCR)

This register is used to control propagation of debug request from DE\_B pad to the cores and debug signals from internal logic to the DE\_B pad.

Address: 0h base + 4h offset = 4h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								DIRECT_ARM_REQ_EN	DIRECT_SDMA_REQ_EN	Reserved	DEBUG_OBS	Reserved	DE_TO_SDMA	DE_TO_ARM	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SJC\_DCR field descriptions**

Field	Description
31–7 -	This field is reserved. Reserved
6 DIRECT_ARM_REQ_EN	Pass Debug Enable event from DE_B pin to ARM platform debug request signal(s). This bit controls the propagation of debug request DE_B to the Arm platform. 0 Disable propagation of system debug to (DE_B pin) to Arm platform. 1 Enable propagation of system debug to (DE_B pin) to Arm platform.
5 DIRECT_SDMA_REQ_EN	Debug enable of the sdma debug request This bit controls the propagation of debug request DE_B to the sdma. 0 Disable propagation of system debug to (DE_B pin) to sdma. 1 Enable propagation of system debug to (DE_B pin) to sdma.
4 -	This field is reserved. Reserved

*Table continues on the next page...*

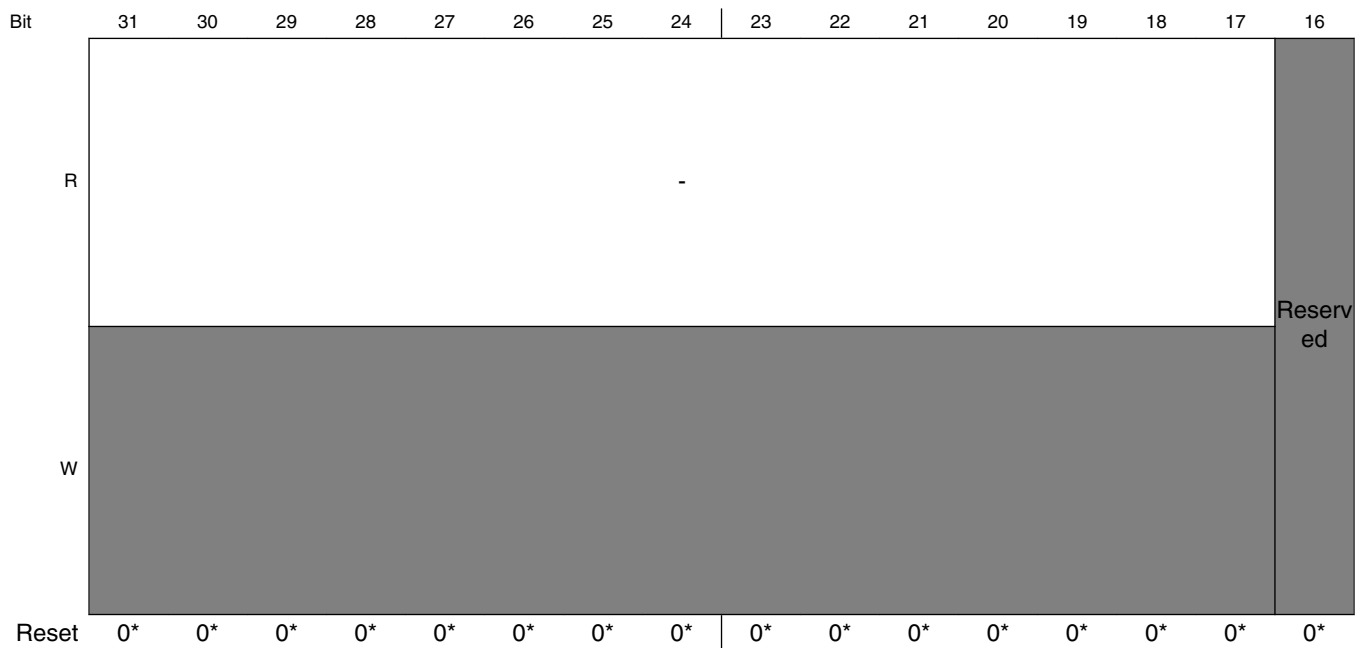
**SJC\_DCR field descriptions (continued)**

Field	Description
3 DEBUG_OBS	<p>Debug observability</p> <p>This bit controls the propagation of the "system debug" input to SJC</p> <p>For i.MX 6x, the SJC's "system_debug" input is tied to logic HIGH value, therefore, set of "debug_obs" bit, will result in unconditional assertion of DE_B pad.</p> <p>0   Disable propagation of system debug to DE_B pin 1   unconditional assertion of pad. DE_B</p>
2 -	<p>This field is reserved.</p> <p>Reserved</p>
1 DE_TO_SDMA	<p>SDMA debug request input propagation</p> <p>This bit controls the propagation of debug request to SDMA, when the JTAG state machine is put in "ENTER_DEBUG" IR instruction..</p> <p>0   Disable propagation of debug request to SDMA 1   Enable propagation of debug request to SDMA</p>
0 DE_TO_ARM	<p>ARM platform debug request input propagation</p> <p>This bit controls the propagation of debug request to ARM platform ("dbgreq"), when the JTAG state machine is put in "ENTER_DEBUG" IR instruction.</p> <p>0   Disable propagation of debug request to ARM platform 1   Enable propagation of debug request to ARM platform</p>

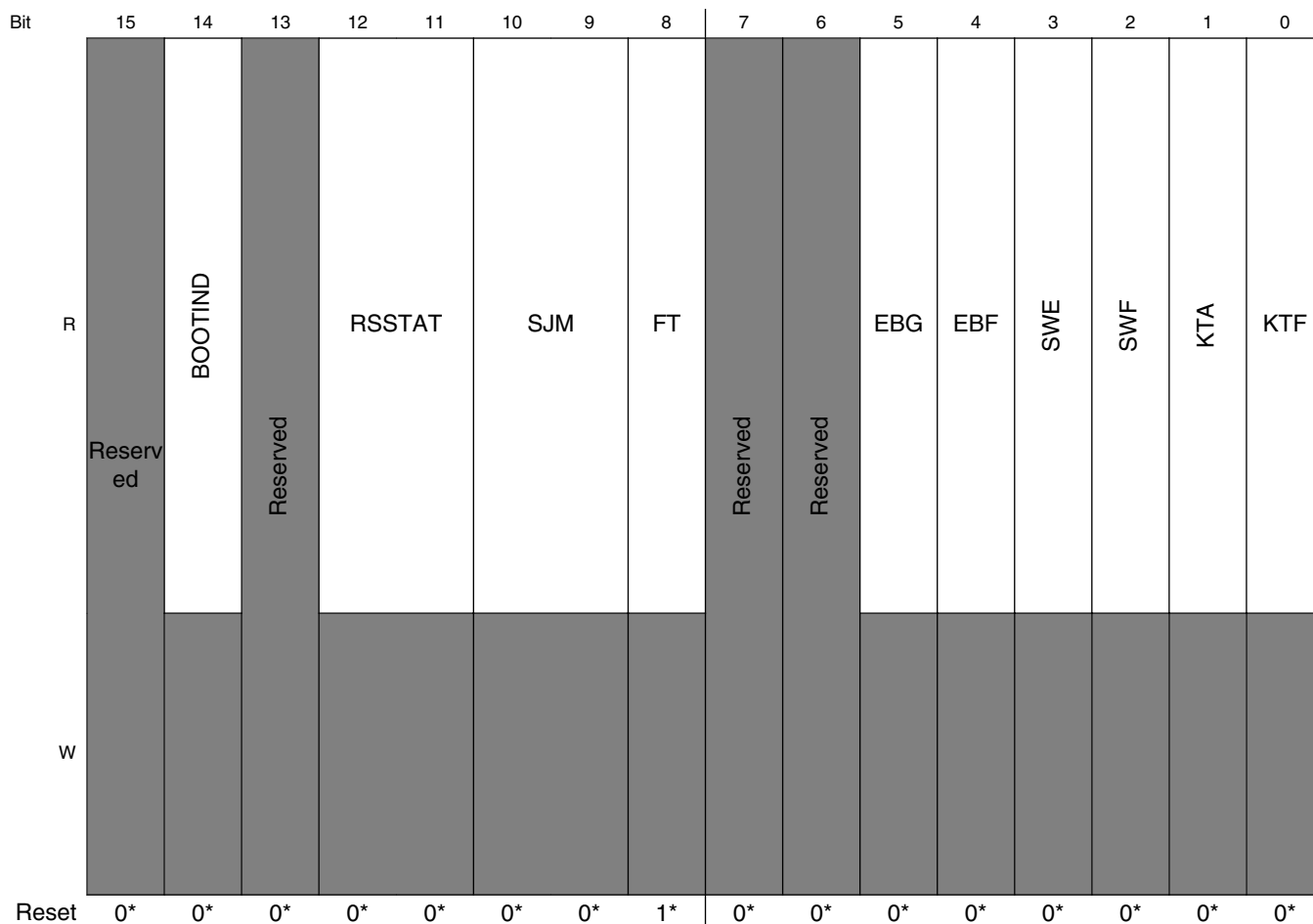


## 43.9.6 Security Status Register (SJC\_SSR)

Address: 0h base + 5h offset = 5h



## SJC Memory Map/Register Definition



\* Notes:

- The SJM reset value, reflects the JTAG security state, as defined by status of JTAG\_SMODE[1:0] fuses. See the [SJM](#) bitfield description for details on valid values.

### SJC\_SSR field descriptions

Field	Description
31–17 -	Reserved.
16–15 -	This field is reserved. Reserved
14 BOOTIND	Boot Indication Inverted Internal Boot indication, i.e inverse of SRC: "src_int_boot" signal
13 -	This field is reserved. Reserved
12–11 RSSTAT	Response status Response status bits  00 Response wasn't entered 01 Response was entered but not verified

Table continues on the next page...

**SJC\_SSR field descriptions (continued)**

Field	Description
	10 Response was entered and is incorrect 11 Response is correct
10–9 SJM	SJC Secure mode Secure JTAG mode, as set by external fuses.  00 No debug (#1) 01 Secure JTAG (#2) 10 Reserved 11 JTAG enabled (#3)
8 FT	Fuse type Fuse type bit - e-fuse or laser fuse  0 E-fuse technology 1 Laser fuse technology
7 -	This field is reserved. Reserved
6 -	This field is reserved. Reserved
5 EBG	External boot granted External boot enabled, requested and granted  1 granted 0 not granted
4 EBF	External Boot fuse Status of the external boot disable fuse  0 (intact) - external boot is allowed 1 (burned) - external boot is disabled
3 SWE	SW enable SW JTAG enable status  1 enabled 0 disabled
2 SWF	Software JTAG enable fuse Status of the no SW disable JTAG fuse  0 (intact) - SW enable possible 1 (intact) - no SW enable possible
1 KTA	Kill Trace is active  1 active 0 not active
0 KTF	Kill Trace Enable fuse value  0 (intact) - kill trace is never active 1 (burned) - kill trace functionality enabled

### 43.9.7 General Purpose Clocks Control Register (SJC\_GPCCR)

This register is used to configure clock related modes in SOC, see System Configuration chapter for more information. Those bits are directly connected to JTAG outputs. Bit 0 of GPCCR controls SDMA clocks invocation. When out of reset, the SDMA is in sleep mode with no SDMA clock running. Unlike events, debug requests does not wake SDMA if it is in sleep mode. The debug request is recognized by the SDMA only when it exits sleep mode upon reception of an event. To be able to enter debug mode even if no event is triggered, the SDMA clock on bit needs to be set prior to sending the debug request (clear at reset).

Address: 0h base + 7h offset = 7h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	-															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	-														ACLKOFFDIS	SCLKR
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

SJC\_GPCCR field descriptions

Field	Description
31–2 -	Reserved
1 ACLKOFFDIS	Disable/prevent ARM platform clock/power shutdown
0 SCLKR	SDMA Clock ON Register - This bit forces the clock on of the SDMA



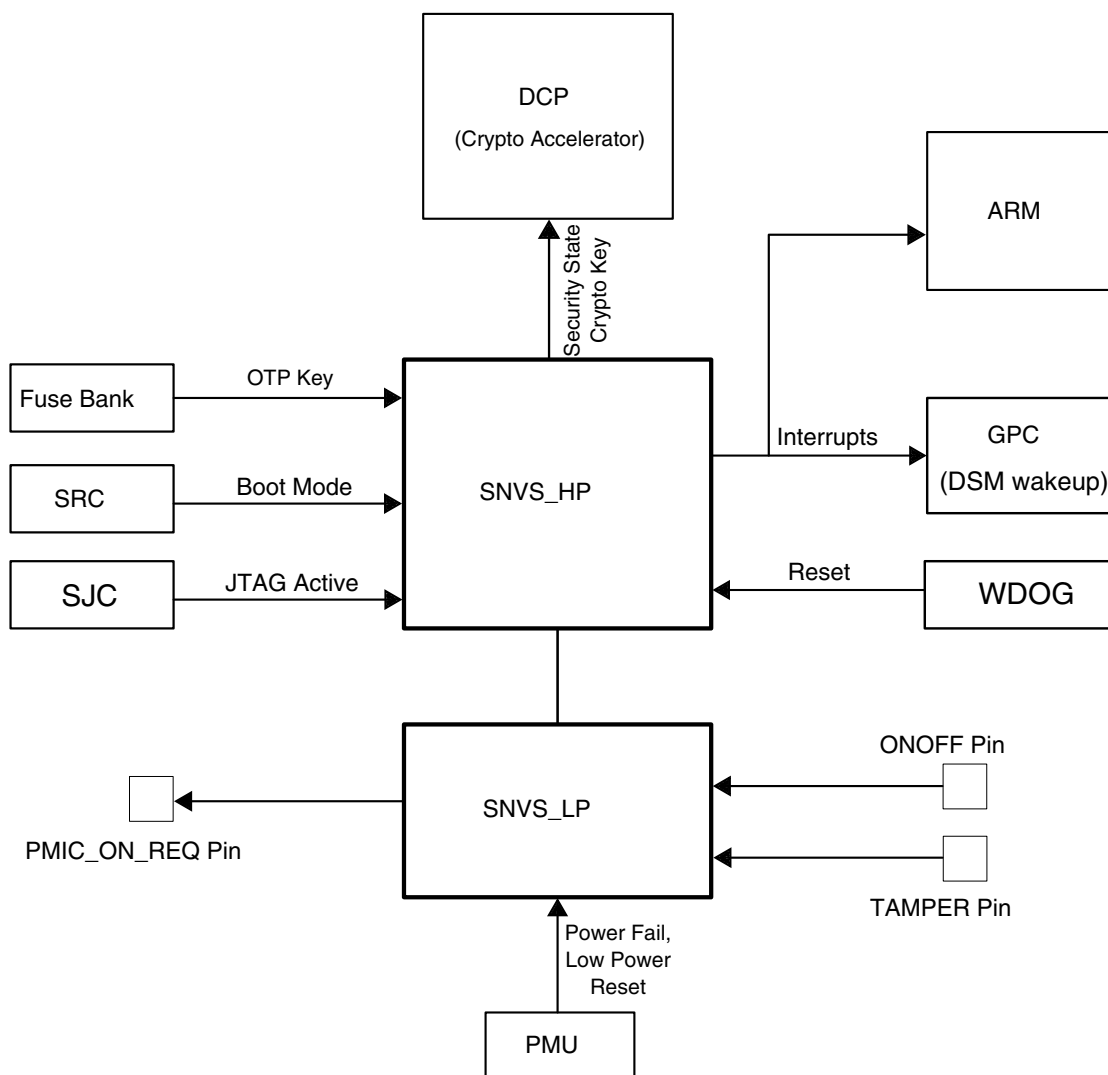
## **Chapter 44**

# **Secure Non-Volatile Storage (SNVS)**

### **44.1 SNVS overview**

This block provides secure non-volatile data storage by means of battery-backed internal registers.

The low-power (battery-backed) section incorporates a secure real time counter, a monotonic counter, and a general purpose register.



**Figure 44-1. Example SNVS Connectivity**

### NOTE

For the security features of SNVS, see the *Security Reference Manual for i.MX 6Dual, 6Quad, 6Solo, and 6DualLite Families of Application Processors (IMX6DQ6SDLRM)* or the *Applications Processor Security Reference Manual for i.MX 6SoloLite (IMX6SLSRM)*.

## 44.1.1 SNVS features

The following table summarizes the features:

**Table 44-1. SNVS feature summary**

Feature	What it does
Secure real time counter (SRTC)	<ul style="list-style-type: none"> <li>The SRTC state is nonvolatile</li> <li>The counter is driven by a dedicated clock, which should always be functional independent of the chip configuration and main chip power state.</li> <li>The counter is a non-rollover counter</li> <li>Programmable time alarm interrupt. <ul style="list-style-type: none"> <li>This alarm generates an interrupt to alert the processor when the time alarm is enabled and the system is powered up.</li> <li>This alarm generates a wake-up alarm via an external pin when the time alarm is enabled, the wake-up external alarm is enabled, and the system is powered down.</li> </ul> </li> <li>The time value is invalidated in case of a security violation.</li> </ul>
Non-secure Real Time Counter	<ul style="list-style-type: none"> <li>The counter is driven by a dedicated clock, which is off when the system power is down.</li> <li>The counter can be synchronized to the value of the Secure Real Time Counter.</li> <li>Programmable time alarm interrupt</li> <li>Periodic interrupt can be generated with different frequencies.</li> </ul>
Real time counter (RTC)	<ul style="list-style-type: none"> <li>The counter is driven by a dedicated clock, which is off when the system power is down</li> <li>The counter can be synchronized to the value of the Secure Real Time Counter</li> <li>Programmable time alarm interrupt</li> <li>Periodic interrupt can be generated with different frequencies</li> </ul>
Monotonic counter	<ul style="list-style-type: none"> <li>The monotonic counter state is nonvolatile.</li> <li>The counter can only increment.</li> <li>The counter is a non-rollover counter</li> <li>The counter value is invalidated in case of security violation.</li> </ul>
General-purpose register	<ul style="list-style-type: none"> <li>The general-purpose register state is nonvolatile.</li> </ul>
Register access protection	<ul style="list-style-type: none"> <li>Privileged software access policy</li> <li>Registers can be programmed only when the system security monitor is in functional state.</li> <li>Some registers/values can only be programmable once per boot cycle.</li> </ul>

## 44.1.2 Modes of operation

The SNVS operates in either the system power-down or system power-up mode of operation.

During system power-down, SNVS\_HP is powered-down. SNVS\_LP is powered from the backup power supply and is electrically isolated from the rest of the chip. In this mode, SNVS\_LP keeps its registers' values .

During system power-up, SNVS\_HP and SNVS\_LP are both powered-up and all SNVS functions are operational.

## 44.2 External Signals

The table found here describes the external signals of SNVS.

**Table 44-2. SNVS External Signals**

Signal	Description	Pad	Mode	Direction
SNVS_PMIC_ON_REQ	Wake-up signal	PMIC_ON_REQ	Not multiplexed	O
SNVS_TAMPER	Tamper signal	TAMPER	Not multiplexed	I

## 44.3 Clocks

The table found here describes the clock sources for SNVS.

Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 44-3. SNVS Clocks**

Clock name	Clock Root	Description
hp_ipg_clk	ipg_clk_root	HP peripheral clock
hp_ipg_clk_s	ipg_clk_root	HP peripheral access clock used for clocking the registers on bus R/W accesses
ipg_hp_rtc_clk	ckil_sync_clk_root	HP RTC clock advances the RTC, doesn't have to be synchronous with any module clock.
lp_ipg_clk	ipg_clk_root	LP peripheral clock
lp_ipg_clk_s	ipg_clk_root	LP peripheral access clock used for register R/W accesses

## 44.4 SNVS structure

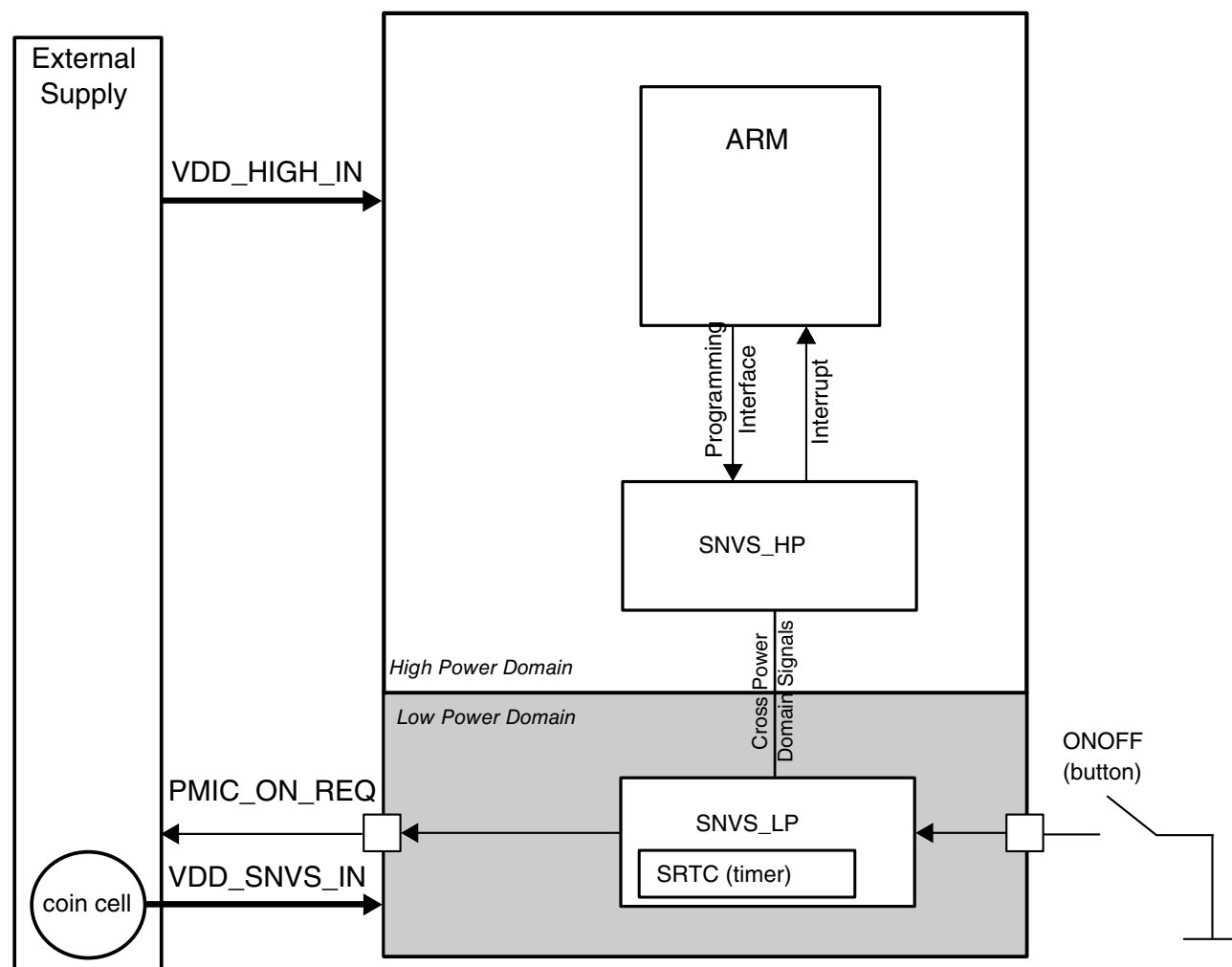
The block is divided into two major submodules based on power supply: the high power domain (SNVS\_HP) and the low power domain (SNVS\_LP).

They are powered as follows:

- SNVS\_LP- dedicated always-powered-on domain
- SNVS\_HP - system (chip) power domain

The following figure illustrates the low power and chip power domains.





**Figure 44-2. SNVS Power Domains**

SNVS\_HP implements all features that enable system communication and provisioning of the SNVS\_LP module.

SNVS\_LP provides hardware that enables secure storage and protection of sensitive data.

#### 44.4.1 SNVS\_HP (high power domain)

SNVS\_HP is partitioned into the following functional units:

- IP bus interface
- SNVS\_LP interface
- Real time counter with alarm
- Control and status registers

SNVS\_HP is in the chip's power supply domain and thus receives power along with the rest of the chip. SNVS\_HP provides an interface between SNVS\_LP and the rest of the system; there is no way to access the SNVS\_LP registers except through the SNVS\_HP. For access to the SNVS\_LP registers, SNVS\_HP must be powered up. It uses a register access permission policy to determine whether access to particular registers is permitted.

## 44.4.2 Non-secure real time counter

SNVS\_HP has an autonomous non-secure real time counter. The counter is not active and is reset when the system is powered down. The HP RTC can be used by any application; it has no privileged software access restrictions. The counter can be synchronized with the SNVS\_LP SRTC by writing to a specific bit in the SNVS\_HP Control Register.

### 44.4.2.1 Calibrating the time counter

The RTC accuracy may suffer from a drift in the clock, which is used to increment the RTC register. To compensate for this drift, a clock calibration mechanism can adjust the RTC value. It is up to the system processor to decide whether calibration is required or not. If RTC correction is required, enable the mechanism and set the calibration value in the control register. The calibration value is a 5 bit value including the sign bit, which is implemented in 2's complement.

If the calibration mechanism is enabled, the calibration value is added or subtracted from the RTC on a periodic basis, once per 32768 cycles of the RTC clock.

The following table shows the available correction range.

**Table 44-4. Time counter calibration settings**

Calibration value setting	Correction in counts per 32768 cycles of the counter clock
01111	+15
:	:
00010	+2
00001	+1
00000	0
11111	-1
11110	-2
:	:
10001	-15
10000	-16

### 44.4.2.2 Time counter alarm

The SNVS\_HP non-secure RTC has its own time alarm register. Any application can update this register. The SNVS\_HP time alarm can generate interrupts to alert the host processor and can wake up the host processor from one of its low-power modes (wait and, doze, stop). Note that this alarm cannot wake up the entire system if it is powered off because this alarm would also be powered off.

### 44.4.2.3 Periodic interrupt

The SNVS\_HP non-secure RTC incorporates a periodic interrupt. The periodic interrupt is generated when a zero-to-one or one-to-zero transition occurs on the selected bit of the RTC. The periodic interrupt source is chosen from 16 bits of the HP RTC according to the PI\_FREQ field setting in the HP Control Register. This bit selection also defines the frequency of the periodic interrupt.

The following figure shows the SNVS\_HP RTC and its interrupts.

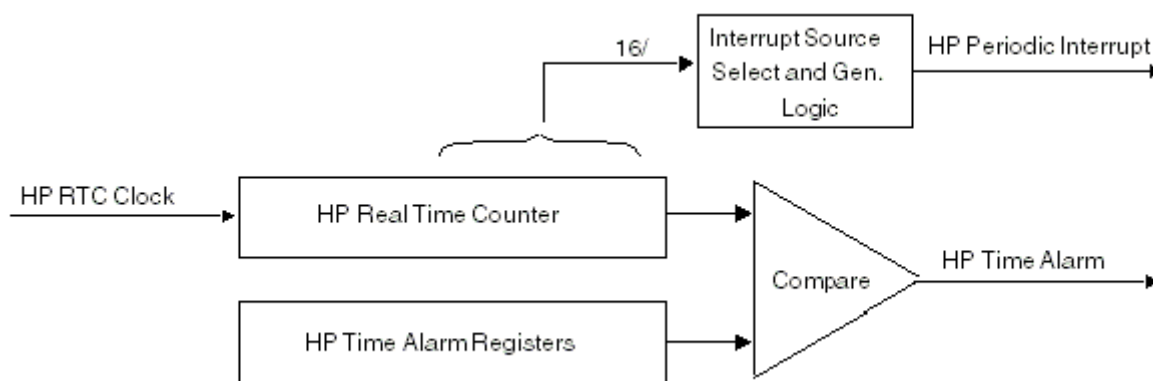


Figure 44-3. SNVS\_HP RTC, alarm, and interrupts

## 44.5 SNVS\_LP (low power domain)

SNVS\_LP has the following functional units:

- Secure non-rollover real time counter with alarm
- Non-rollover monotonic counter
- General purpose register
- Control and status registers

The SNVS\_LP is a data storage subsystem with enhanced security capabilities. Its purpose is to store and protect system secure data, regardless of the main system power state.

SNVS\_LP is in the always-powered-up domain, which is a separate power domain with its own power supply.

### **44.5.1 Behavior during system power down**

When the chip power supply domain loses power, SNVS\_LP continues to operate normally, and it ignores all inputs from SNVS\_HP.

### **44.5.2 Secure real time counter (SRTC)**

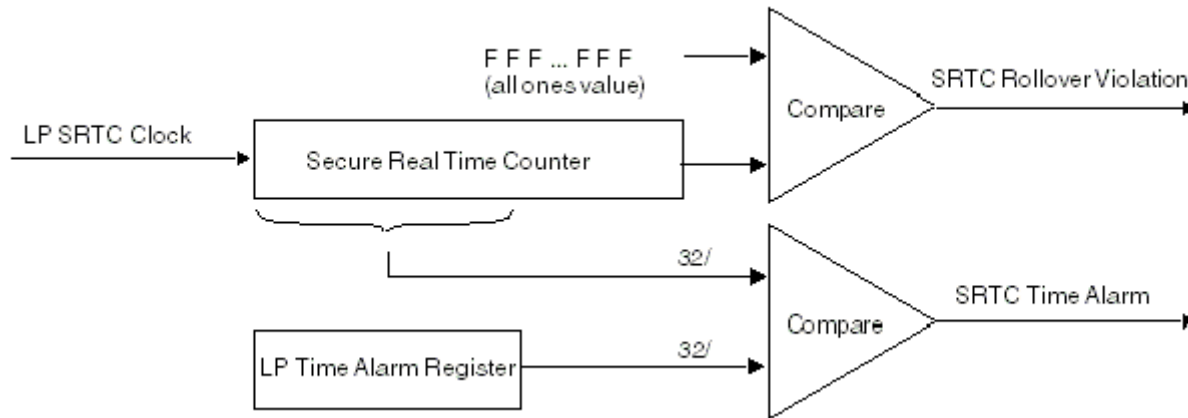
The SNVS\_LP incorporates an autonomous SRTC. This is a non-rollover counter, which means that the SRTC does not rollover when it reaches the maximum value of all ones. Instead, a time rollover indication is generated to the SNVS\_LP tamper monitor, which generates a security violation and interrupt.

#### **44.5.2.1 Calibrating the SRTC time counter**

To compensate for possible drift in the SRTC clock source, use the clock calibration mechanism implemented with the SRTC. This mechanism works the same way as the non-secure RTC clock calibration mechanism. Refer to the [Calibrating the time counter](#) for the detailed description of its functionality.

#### **44.5.2.2 Time counter alarm (zmk)**

The following figure shows the SNVS\_LP Secure Real Time Counter, time alarms, and rollover security violation.

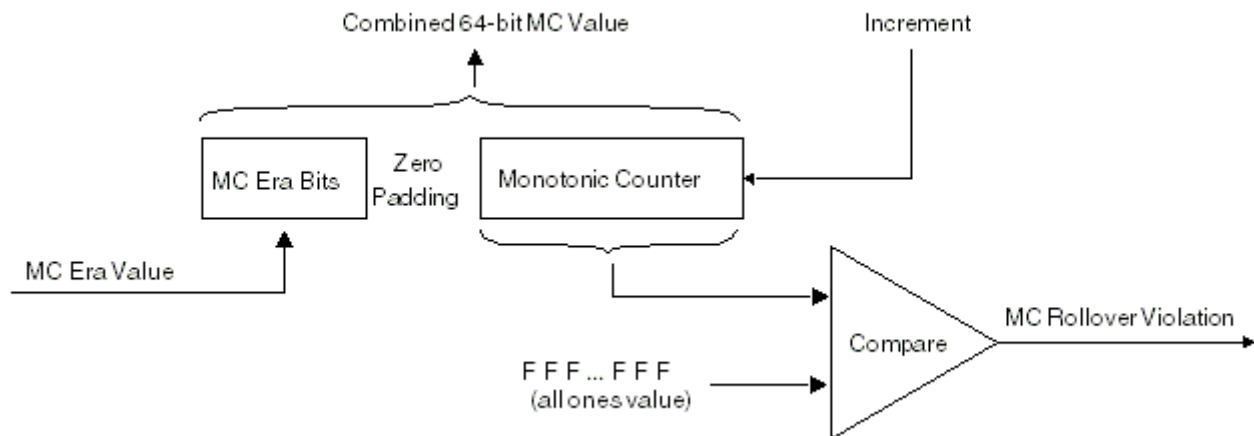


**Figure 44-4. SNVS\_LP secure real time counter**

SNVS\_LP has its own 32-bit LP Time Alarm register. As illustrated in [Figure 44-4](#), this register generates an SRTC time alarm once the secure real time clock's 32 most significant bits match with the LP Time Alarm register. The time alarm can generate an interrupt to alert the host processor and can wake the host processor from one of its low-power modes (Wait and, Doze, Stop). This alarm can also wake up the entire system in the power-down mode by asserting the wake-up external output signal.

### 44.5.3 Monotonic counter (MC)

The following figure shows the MC and its rollover security violation.



**Figure 44-5. SNVS\_LP monotonic counter**

Some security applications require a monotonic counter (MC) that cannot be exhausted or returned to any previous value during the product's lifetime. Because the MC can never repeat a number, it cannot be reset or cycled back to its starting count. If it reaches its

maximum value, it does not rollover. Instead, a monotonic counter rollover indication is generated to the SNVS \_LP tamper monitor. This generates an interrupt to the host processor.

The SNVS uses an ERA value derived from the OTP elements as a mechanism for recovery from an MC failure (for example, due to a failure of LP power) where the MC value was compromised or cleared. The ERA value is prepended to the MC to form its most significant bits. Once any of the ERA value bits are set, the MC can count up from any value, including zero. This guarantees that any future value of the combined monotonic counter will be greater than any of its past values.

## 44.6 SNVS reset and system power up

The table found here describes information about reset summary for SNVS.

**Table 44-5. Reset summary**

Reset	Source	Characteristics	Internally resets
HP Hard	ipg_hard_async_reset_b	active-low, asynchronous	All SNVS_HPSNVS_LP registers and flops.
LP Power On Reset (POR)	lp_por_b	active-low, asynchronous	All SNVS_LP registers and flops
Field Return System Configuration	System Security Configuration Input	active-high	This reset is permanently active in Field Return Configuration
LP software Reset	software	active-high, synchronous, 1 cycle	All SNVS_LP registers and flops. LP software Reset can be asserted if not disabled.

### 44.6.1 PMIC Interface

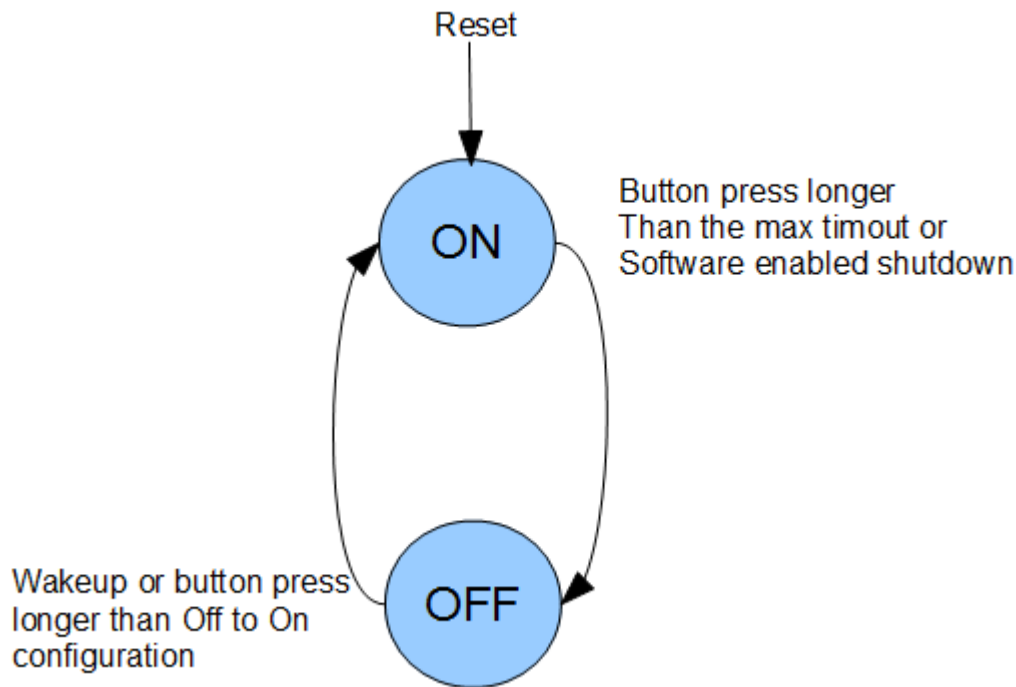
The On/Off logic inside of SNVS\_LP allows for connecting directly to a PMIC or other voltage regulator device. The logic takes a button input signal and then outputs a pmic\_en\_b and set\_pwr\_off\_irq. PMIC logic also supports the SNVS\_LP tamper logic which will allow waking the system up when a tamper event has happened while in the OFF state. The logic has two different modes of operation (Dumb and Smart mode).

Dumb PMIC Mode:

The dumb pmic mode uses pmic\_en\_b to issue a level signal for on and off. Dumb pmic mode has many different configuration options which include (debounce, off to on time, and max time out).

- **Debounce:** The debounce configuration supports 0 msec, 50 msec, 100 msec and 500 msec. The debounce is used to generate the set\_pwr\_off\_irq interrupt. While in the ON state and the button is pressed longer than the debounce time the set\_pwr\_off\_irq is generated.
- **Off to On Time:** The Off to On configuration supports 0 msec, 50 msec, 100 msec, and 500 msec. This configuration supports the time it takes to request power on after the configured button press time has been reached. Once the button is pressed longer than the configuration time, the state machine will transition from the OFF to the ON state.
- **Max Timeout:** The max timeout configuration supports 5 secs, 10 secs, 15 secs and disable. This configuration supports the time it takes to request power down after the button has been pressed for the defined time.

The dumb PMIC mode uses a 2 state state machine, as shown below. The output of the pmic\_en\_b is generated by the state of the state machine.



Smart PMIC Mode:

The smart PMIC mode is meant to connect to another PMIC. The `pmic_en_b` signal issues a pulse instead of a level signal. The only configuration option available for this mode is the Debounce configuration which is used for the `set_pwr_off_irq`.

## 44.7 SNVS interrupts, alarms, and security violations

SNVS provides the following interrupt and alarm lines:

- Functional interrupt (active low)
- Wake-up alarm
- Real-time clock period interrupt
- Power off (button) interrupt

The following table summarizes all SNVS interrupts and alarm sources.

**Table 44-6. Interrupts and alarms summary**

Interrupt/violation	Source	Default configuration <sup>1</sup>	Configuration options
SNVS functional interrupt	SRTC time alarm	Disable	Enable/Disable
	RTC time alarm	Disable	Enable/Disable
	RTC periodic interrupt	Disable	Enable/Disable
SNVS power off (button) interrupt	BTN input signal	50msec debounce	Debounce time
SNVS wake-up alarm <sup>2</sup>	SRTC time alarm	Disable	Enable/Disable
	LP security violation	Disable	Enable/Disable

1. Default behavior refers to the setting after LP/HP reset.

2. This alarm is functional only in the system power-down mode when the LP section is isolated.

## 44.8 Programming Guidelines

This section provides initialization and application information for the SNVS module.

### 44.8.1 RTC/SRTC control bits setting

All SNVS registers are programmed from the register bus. Therefore, any changes are synchronized with the IP clock. Several registers can also change synchronously with the RTC/SRTC clock after they are programmed. To avoid IP clock and RTC/SRTC clock



synchronization issues, these values can only be programmed when the corresponding function is disabled. The following table presents the list of these values with the control bit setting required for programming.

**Table 44-7. RTC/SRTC synchronized values list**

Function	Value/register	Control bit setting
HP section		
HP Real Time Counter	HPRTCMR and HPRTCLR Registers	RTC_EN = 0 - HPRTCMR/HPRTCLR can be programmed RTC_EN = 1 - HPRTCMR/HPRTCLR cannot be programmed
HP Time Alarm	HPTAMR and HPTALR Registers	HPTA_EN = 0 - HPTAMR/HPTALR can be programmed HPTA_EN = 1 - HPTAMR/HPTALR cannot be programmed
HP Time Calibration Value	HPCALB_VAL Value	HPCALB_EN = 0 - HPCALB_VAL can be programmed HPCALB_EN = 1 - HPCALB_VAL cannot be programmed
LP section		
LP Secure Real Time Counter	LPRTCMR and LPRTCLR Registers	SRTC_ENV = 0 - LPRTCMR/LPRTCLR can be programmed SRTC_ENV = 1 - LPRTCMR/LPRTCLR cannot be programmed
LP Time Alarm	LPTAR Register	LPTA_EN = 0 - LPTAR can be programmed LPTA_EN = 1 - LPTAR cannot be programmed
LP Time Calibration Value	LPCALB_VAL Value	LPCALB_EN = 0 - LPCALB_VAL can be programmed LPCALB_EN = 1 - LPCALB_VAL cannot be programmed

Use the following step to program synchronized values:

1. Check the enable bit value. If set, clear it.
2. Verify that the enable bit is cleared.

There are two reasons to verify the enable bit's setting:

- Enable bit clearing does not happen immediately; it takes three IPclock cycles and two RTC/SRTC clock cycles to change the enable bit's value.
  - If the enable bit is locked for programming, it cannot be cleared.
3. Program the desired value.
  4. Set the enable bit; it takes three IP clock cycles and two RTC/SRTC clock cycles for the bit to set.

### NOTE

Incrementing the value programmed into RTC/SRTC registers by two compensates for the two RTC/SRTC clock cycle delay that is required to enable the counter.

## 44.8.2 RTC/SRTC value read

There are two scenarios when software can read corrupted values from the RTC (HPRTCMR and HPRTCLR) and SRTC (LPSRTCMR and LPSRTCLR) registers:

- The RTC and SRTC counters are incremented by the slow 32 kHz clock, which is asynchronous to the system clock. The counter value is synchronized to the system clock before software reads that. The synchronization register may capture the counter value in the middle of the counter update. In this case, it is not guaranteed that all bits are properly sampled by the synchronization register; the value read by software can be wrong.
- The RTC and SRTC value is longer than the single bus read transaction of 32-bits. Therefore, software reads two registers, each holding a portion of the counter value. After reading one of these registers but before reading the second register, both registers may update their values. In this case, the value combined by software will be incorrect.

To avoid these issues, it is strongly recommended that software perform two consecutive reads of the RTC/SRTC value:

- If two consecutive reads are similar, the value is correct.
- If two consecutive reads are different, perform two more reads.

The worst case scenario may require three sessions of two consecutive reads.

## 44.8.3 General initialization guidelines

Complete the following steps in order to properly initialize the module:

1. Enable interrupts in SNVS control and configuration registers.
2. Program SNVS general functions/configurations.
3. User Specific: Set lock bits.

## 44.9 SNVS Memory Map/Register Definition

This section contains detailed register descriptions for the SNVS registers. Each description includes a standard register diagram and register table. The register table provides detailed descriptions of the register bit and field function, in bit order.

SNVS registers consist of two types:

- Privileged read/write accessible
- Non-privileged read/write accessible

Privileged read/write accessible registers can only be accessed for read/write by privileged software. Unauthorized write accesses are ignored, and unauthorized read accesses return zero. Non-privileged software can access privileged access registers when the non-privileged software access enable bit is set in the SNVS\_HP Command Register.

- Non-Secure
- Trusted
- Secure

Non-privileged read/write accessible registers are read/write accessible by any software.

The following table shows the SNVS memory map. The LP register values are set only on LP POR and are unaffected by System (HP) POR. The HP registers are set only on System POR and are unaffected by LP POR.

### NOTE

For more information on security-related bitfields, see the *Security Reference Manual for i.MX 6Dual, 6Quad, 6Solo, and 6DualLite Families of Application Processors (IMX6DQ6SDL SRM)* or the *Applications Processor Security Reference Manual for i.MX 6SoloLite (IMX6SL SRM)*.

### SNVS memory map

Offset address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20C_C000	SNVS_HP Lock Register (SNVS_HPLR)	32	R/W	0000_0000h	<a href="#">44.9.1/2737</a>
20C_C004	SNVS_HP Command Register (SNVS_HPCOMR)	32	R/W	0000_0000h	<a href="#">44.9.2/2739</a>
20C_C008	SNVS_HP Control Register (SNVS_HPCR)	32	R/W	0000_0000h	<a href="#">44.9.3/2741</a>
20C_C014	SNVS_HP Status Register (SNVS_HPSR)	32	R/W	<a href="#">See section</a>	<a href="#">44.9.4/2744</a>
20C_C024	SNVS_HP Real Time Counter MSB Register (SNVS_HPRTCMR)	32	R/W	0000_0000h	<a href="#">44.9.5/2746</a>
20C_C028	SNVS_HP Real Time Counter LSB Register (SNVS_HPRTCLR)	32	R/W	0000_0000h	<a href="#">44.9.6/2747</a>
20C_C02C	SNVS_HP Time Alarm MSB Register (SNVS_HPTAMR)	32	R/W	0000_0000h	<a href="#">44.9.7/2747</a>
20C_C030	SNVS_HP Time Alarm LSB Register (SNVS_HPTALR)	32	R/W	0000_0000h	<a href="#">44.9.8/2748</a>
20C_C034	SNVS_LP Lock Register (SNVS_LPLR)	32	R/W	0000_0000h	<a href="#">44.9.9/2749</a>
20C_C038	SNVS_LP Control Register (SNVS_LPCR)	32	R/W	0000_0000h	<a href="#">44.9.10/2751</a>

Table continues on the next page...

## SNVS memory map (continued)

Offset address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20C_C04C	SNVS_LP Status Register (SNVS_LPSR)	32	R/W	0000_0008h	<a href="#">44.9.11/2754</a>
20C_C050	SNVS_LP Secure Real Time Counter MSB Register (SNVS_LPSRTCMBR)	32	R/W	0000_0000h	<a href="#">44.9.12/2756</a>
20C_C054	SNVS_LP Secure Real Time Counter LSB Register (SNVS_LPSRTCLR)	32	R/W	0000_0000h	<a href="#">44.9.13/2756</a>
20C_C058	SNVS_LP Time Alarm Register (SNVS_LPTAR)	32	R/W	0000_0000h	<a href="#">44.9.14/2757</a>
20C_C05C	SNVS_LP Secure Monotonic Counter MSB Register (SNVS_LPSMCMR)	32	R/W	0000_0000h	<a href="#">44.9.15/2757</a>
20C_C060	SNVS_LP Secure Monotonic Counter LSB Register (SNVS_LPSMCLR)	32	R/W	0000_0000h	<a href="#">44.9.16/2758</a>
20C_C068	SNVS_LP General Purpose Register (SNVS_LPGPR)	32	R/W	0000_0000h	<a href="#">44.9.17/2758</a>
20C_CBF8	SNVS_HP Version ID Register 1 (SNVS_HPVIDR1)	32	R	003E_0100h	<a href="#">44.9.18/2759</a>
20C_CBFC	SNVS_HP Version ID Register 2 (SNVS_HPVIDR2)	32	R	0000_0000h	<a href="#">44.9.19/2759</a>

### 44.9.1 SNVS\_HP Lock Register (SNVS\_HPLR)

The SNVS\_HP Lock Register contains lock bits for the SNVS registers. This is a privileged write register.

Address: 20C\_C000h base + 0h offset = 20C\_C000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved													Security	HPSICR_L	HPSVCR_L
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved						Security	LPTDCR_SL	Security	Reserved	GPR_SL	MC_SL	LPCALB_SL	SRTC_SL	Security	Security
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SNVS\_HPLR field descriptions**

Field	Description
31–19 -	This field is reserved.
18 Security	Security-related field.
17 HPSICR_L	HP Security Interrupt Control Register Lock When set, prevents any writes to the HPSICR. Once set, this bit can only be reset by the system reset.  0 Write access is allowed 1 Write access is not allowed
16 HPSVCR_L	HP Security Violation Control Register Lock When set, prevents any writes to the HPSVCR. Once set, this bit can only be reset by the system reset.

*Table continues on the next page...*

## SNVS\_HPLR field descriptions (continued)

Field	Description
	0 Write access is allowed 1 Write access is not allowed
15–10 -	This field is reserved.
9 Security	Security-related field.
8 LPTDCR_SL	LP Tamper Detectors Configuration Register Soft Lock When set, prevents any writes to the LPTDCR. Once set, this bit can only be reset by the system reset.  0 Write access is allowed 1 Write access is not allowed
7 Security	Security-related field.
6 -	This field is reserved.
5 GPR_SL	General Purpose Register Soft Lock When set, prevents any writes to the GPR. Once set, this bit can only be reset by the system reset.  0 Write access is allowed 1 Write access is not allowed
4 MC_SL	Monotonic Counter Soft Lock When set, prevents any writes (increments) to the MC Registers and MC_ENV bit. Once set, this bit can only be reset by the system reset.  0 Write access (increment) is allowed 1 Write access (increment) is not allowed
3 LPCALB_SL	LP Calibration Soft Lock When set, prevents any writes to the LP Calibration Value (LPCALB_VAL) and LP Calibration Enable (LPCALB_EN). Once set, this bit can only be reset by the system reset.  0 Write access is allowed 1 Write access is not allowed
2 SRTC_SL	Secure Real Time Counter Soft Lock When set, prevents any writes to the SRTC Registers, SRTC_ENV, and SRTC_INV_EN bits. Once set, this bit can only be reset by the system reset.  0 Write access is allowed 1 Write access is not allowed
1 Security	Security-related field.
0 Security	Security-related field.

## 44.9.2 SNVS\_HP Command Register (SNVS\_HPCOMR)

The SNVS\_HP Command Register contains the command, configuration, and control bits for the SNVS block. This is a privileged write register.

Address: 20C\_C000h base + 4h offset = 20C\_C004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	NPSWA_EN	Reserved												Security	Security	Security
W														Security	Security	Security
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved		Security	Security			Security	Security	Reserved		LP_SWR_DIS	LP_SWR	Reserved	Security	Security	Security
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## SNVS\_HPCOMR field descriptions

Field	Description
31 NPSWA_EN	Non-Privileged Software Access Enable When set, allows non-privileged software to access all SNVS registers, including those that are privileged software read/write access only. 0 Only privileged software can access privileged registers 1 Any software can access privileged registers
30–20 -	This field is reserved.
19 Security	Security-related field.
18 Security	Security-related field.
17 Security	Security-related field.
16 Security	Security-related field.
15–14 -	This field is reserved.
13 Security	Security-related field.
12–11 Security	Security-related field.
10 Security	Security-related field.
9 Security	Security-related field.
8 Security	Security-related field.
7–6 -	This field is reserved.
5 LP_SWR_DIS	LP Software Reset Disable When set, disables the LP software reset. Once set, this bit can only be reset by the system reset. 0 LP software reset is enabled 1 LP software reset is disabled
4 LP_SWR	LP Software Reset When set, it resets the SNVS_LP section. This bit cannot be set when the LP_SWR_DIS bit is set. This self-clearing bit is always read as zero. 0 No Action 1 Reset LP section
3 -	This field is reserved.
2 Security	Security-related field.
1 Security	Security-related field.

*Table continues on the next page...*



**SNVS\_HPCOMR field descriptions (continued)**

Field	Description
0 Security	Security-related field.

**44.9.3 SNVS\_HP Control Register (SNVS\_HPCR)**

The SNVS\_HP Control Register contains various control bits of the HP section of SNVS.

Address: 20C\_C000h base + 8h offset = 20C\_C008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															HP_TS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	HPCALB_VAL						Reserved	PI_FREQ				PI_EN	Reserved	HPTA_EN	RTC_EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SNVS\_HPCR field descriptions**

Field	Description
31–17 -	This field is reserved.
16 HP_TS	HP Time Synchronize When set, this updates the HP Time Counter with the LP Time Counter value. This self-clearing bit is always read as zero.

Table continues on the next page...

## SNVS\_HPCR field descriptions (continued)

Field	Description
	0 No Action 1 Synchronize the HP Time Counter to the LP Time Counter
15 -	This field is reserved.
14–10 HPCALB_VAL	<p>HP Calibration Value</p> <p>Defines signed calibration value for the HP Real Time Counter. This field can be programmed only when RTC Calibration is disabled (HPCALB_EN is not set). This is a 5-bit 2's complement value. Hence, the allowable calibration values are in the range from -16 to +15 counts per 32768 ticks of the counter.</p> <p>00000 +0 counts per each 32768 ticks of the counter  00001 +1 counts per each 32768 ticks of the counter  00010 +2 counts per each 32768 ticks of the counter  01111 +15 counts per each 32768 ticks of the counter  10000 -16 counts per each 32768 ticks of the counter  10001 -15 counts per each 32768 ticks of the counter  11110 -2 counts per each 32768 ticks of the counter  11111 -1 counts per each 32768 ticks of the counter</p>
9 -	This field is reserved.
8 HPCALB_EN	<p>HP Real Time Counter Calibration Enabled</p> <p>Indicates that the time calibration mechanism is enabled.</p> <p>0 HP Timer calibration disabled  1 HP Timer calibration enabled</p>
7–4 PI_FREQ	<p>Periodic Interrupt Frequency</p> <p>Defines frequency of the periodic interrupt. The interrupt is generated when a zero-to-one or one-to-zero transition occurs on the selected bit of the HP Real Time Counter and Real Time Counter and Periodic Interrupt are both enabled (RTC_EN and PI_EN are set). It is recommended to program this field when Periodic Interrupt is disabled (PI_EN is not set). The possible frequencies are:</p> <p>0000 - bit 0 of the RTC is selected as a source of the periodic interrupt  0001 - bit 1 of the RTC is selected as a source of the periodic interrupt  0010 - bit 2 of the RTC is selected as a source of the periodic interrupt  0011 - bit 3 of the RTC is selected as a source of the periodic interrupt  0100 - bit 4 of the RTC is selected as a source of the periodic interrupt  0101 - bit 5 of the RTC is selected as a source of the periodic interrupt  0110 - bit 6 of the RTC is selected as a source of the periodic interrupt  0111 - bit 7 of the RTC is selected as a source of the periodic interrupt  1000 - bit 8 of the RTC is selected as a source of the periodic interrupt  1001 - bit 9 of the RTC is selected as a source of the periodic interrupt  1010 - bit 10 of the RTC is selected as a source of the periodic interrupt  1011 - bit 11 of the RTC is selected as a source of the periodic interrupt  1100 - bit 12 of the RTC is selected as a source of the periodic interrupt  1101 - bit 13 of the RTC is selected as a source of the periodic interrupt  1110 - bit 14 of the RTC is selected as a source of the periodic interrupt  1111 - bit 15 of the RTC is selected as a source of the periodic interrupt</p>

Table continues on the next page...

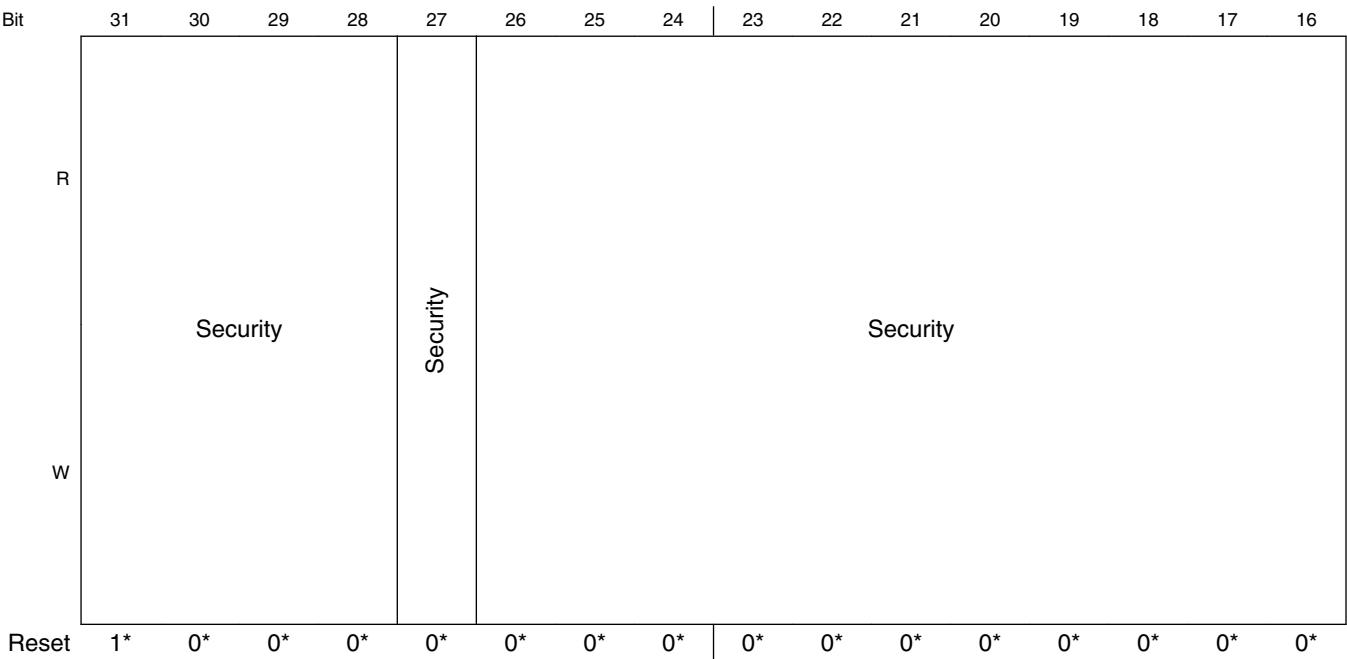
**SNVS\_HPCR field descriptions (continued)**

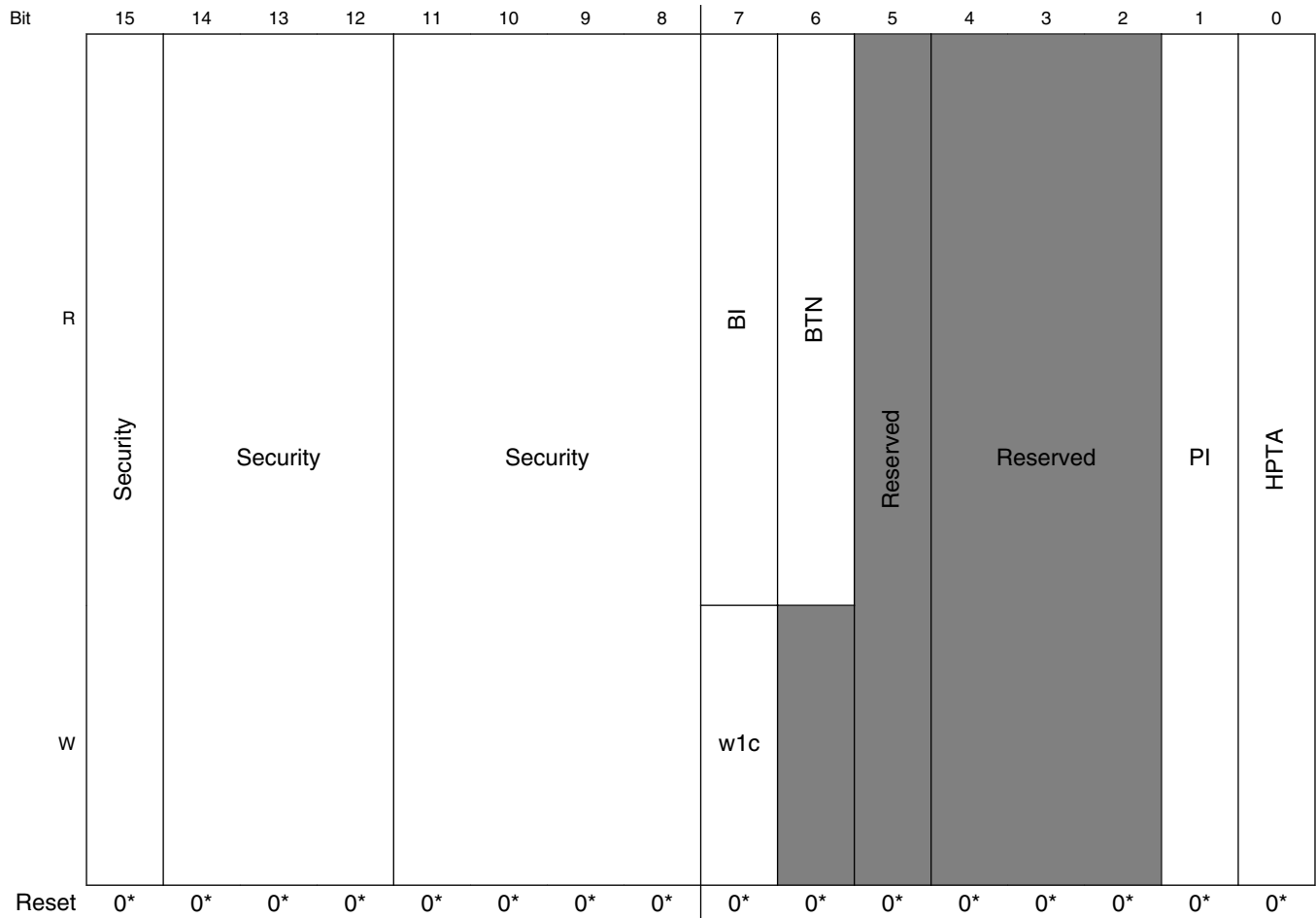
<b>Field</b>	<b>Description</b>
3 PI_EN	<p>HP Periodic Interrupt Enable</p> <p>The periodic interrupt can be generated only if the HP Real Time Counter is enabled.</p> <p>0 HP Periodic Interrupt is disabled 1 HP Periodic Interrupt is enabled</p>
2 -	This field is reserved.
1 HPTA_EN	<p>HP Time Alarm Enable</p> <p>When set, the time alarm interrupt is generated if the value in the HP Time Alarm Registers is equal to the value of the HP Real Time Counter.</p> <p>0 HP Time Alarm Interrupt is disabled 1 HP Time Alarm Interrupt is enabled</p>
0 RTC_EN	<p>HP Real Time Counter Enable</p> <p>0 RTC is disabled 1 RTC is enabled</p>

44.9.4 SNVS\_HP Status Register (SNVS\_HPSR)

The HP Status Register reflects the internal state of the SNVS .

Address: 20C\_C000h base + 14h offset = 20C\_C014h





\* Notes:

- The reset value depends on the value of the OTPMK programmed in fuses

### SNVS\_HPSR field descriptions

Field	Description
31–28 Security	Security-related field.
27 Security	Security-related field.
26–16 Security	Security-related field.
15 Security	Security-related field.
14–12 Security	Security-related field.
11–8 Security	Security-related field.
7 BI	Button Interrupt. Signal ipi_snvs_btn_int_b was asserted.

Table continues on the next page...

**SNVS\_HPSR field descriptions (continued)**

Field	Description
6 BTN	Value of the BTN input. This is the external button used for PMIC control. 1: BTN pressed 0: BTN not pressed
5 -	This field is reserved.
4–2 -	This field is reserved.
1 PI	Periodic Interrupt Indicates that periodic interrupt has occurred since this bit was last cleared.  0 No periodic interrupt occurred. 1 A periodic interrupt occurred.
0 HPTA	HP Time Alarm Indicates that the HP Time Alarm has occurred since this bit was last cleared.  0 No time alarm interrupt occurred. 1 A time alarm interrupt occurred.

**44.9.5 SNVS\_HP Real Time Counter MSB Register (SNVS\_HPRTCMR)**

The SNVS\_HP Real Time Counter MSB register contains the most significant bits of the HP Real Time Counter.

Address: 20C\_C000h base + 24h offset = 20C\_C024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SNVS\_HPRTCMR field descriptions**

Field	Description
31–15 -	This field is reserved. Reserved
14–0 RTC	HP Real Time Counter Most significant 32 bits. This register can be programmed only when RTC is not active (RTC_EN bit is not set).

### 44.9.6 SNVS\_HP Real Time Counter LSB Register (SNVS\_HPRTCLR)

The SNVS\_HP Real Time Counter LSB register contains the 32 least significant bits of the HP real time counter.

Address: 20C\_C000h base + 28h offset = 20C\_C028h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RTC																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SNVS\_HPRTCLR field descriptions**

Field	Description
31–0 RTC	HP Real Time Counter Least significant 32 bits. This register can be programmed only when RTC is not active (RTC_EN bit is not set).

### 44.9.7 SNVS\_HP Time Alarm MSB Register (SNVS\_HPTAMR)

The SNVS\_HP Time Alarm MSB register contains the most significant bits of the SNVS\_HP Time Alarm value.

Address: 20C\_C000h base + 2Ch offset = 20C\_C02Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																HPTA															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

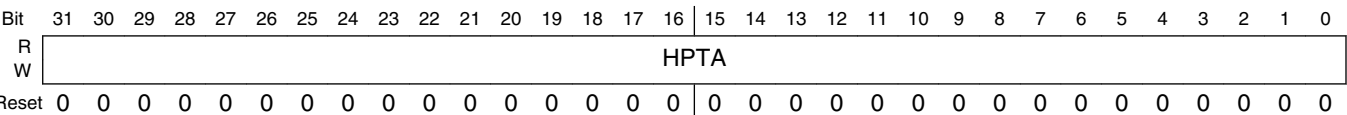
**SNVS\_HPTAMR field descriptions**

Field	Description
31–15 -	This field is reserved.
14–0 HPTA	HP Time Alarm Most significant 15 bits. This register can be programmed only when HP time alarm is disabled (HPTA_EN bit is not set).

### 44.9.8 SNVS\_HP Time Alarm LSB Register (SNVS\_HPTALR)

The SNVS\_HP Time Alarm LSB register contains the 32 least significant bits of the SNVS\_HP Time Alarm value.

Address: 20C\_C000h base + 30h offset = 20C\_C030h



SNVS\_HPTALR field descriptions

Field	Description
31–0 HPTA	HP Time Alarm Least significant bits. This register can be programmed only when HP time alarm is disabled (HPTA_EN bit is not set).



### 44.9.9 SNVS\_LP Lock Register (SNVS\_LPLR)

The SNVS\_LP Lock Register contains lock bits for the SNVS\_LP registers.

Address: 20C\_C000h base + 34h offset = 20C\_C034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved							Security	Reserved	Reserved	GPR_HL	MC_HL	LPCALB_HL	SRTC_HL	Security	Security
W								Security	Reserved	Reserved	GPR_HL	MC_HL	LPCALB_HL	SRTC_HL	Security	Security
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SNVS\_LPLR field descriptions**

Field	Description
31–10 -	This field is reserved.
9 Security	Security-related field.
8 Security	Security-related field.
7 -	This field is reserved.
6 -	This field is reserved.
5 GPR_HL	General Purpose Register Hard Lock When set, prevents any writes to the GPR. Once set, this bit can only be reset by the LP POR.  0 Write access is allowed. 1 Write access is not allowed.

*Table continues on the next page...*

**SNVS\_LPLR field descriptions (continued)**

Field	Description
4 MC_HL	<p>Monotonic Counter Hard Lock</p> <p>When set, prevents any writes (increments) to the MC Registers and MC_ENV bit. Once set, this bit can only be reset by the LP POR.</p> <p>0 Write access (increment) is allowed. 1 Write access (increment) is not allowed.</p>
3 LPCALB_HL	<p>LP Calibration Hard Lock</p> <p>When set, prevents any writes to the LP Calibration Value (LPCALB_VAL) and LP Calibration Enable (LPCALB_EN). Once set, this bit can only be reset by the LP POR.</p> <p>0 Write access is allowed. 1 Write access is not allowed.</p>
2 SRTC_HL	<p>Secure Real Time Counter Hard Lock</p> <p>When set, prevents any writes to the SRTC registers, SRTC_ENV, and SRTC_INV_EN bits. Once set, this bit can only be reset by the LP POR.</p> <p>0 Write access is allowed. 1 Write access is not allowed.</p>
1 Security	Security-related field.
0 Security	Security-related field.

### 44.9.10 SNVS\_LP Control Register (SNVS\_LPCR)

The SNVS\_LP Control Register contains various control bits of the LP section of SNVS.

Address: 20C\_C000h base + 38h offset = 20C\_C038h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								PK_OVERRIDE	PK_EN	ON_TIME		DEBOUNCE		BTN_PRESS_TIME	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0		
R	Reserved	LPCALB_VAL							Reserved	LPCALB_EN	PWR_GLITCH_EN	TOP	DP_EN	SRTC_INV_EN	LPWUI_EN	MC_ENV	LPTA_EN	SRTC_ENV
W																		
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		

**SNVS\_LPCR field descriptions**

Field	Description
31–24 -	This field is reserved.
23 PK_OVERRIDE	PMIC On Request Override. The value written to PK_OVERRIDE will be asserted on output signal snvs_lp_pk_override. That signal is used to override the IOMUX control for the PMIC I/O pad.
22 PK_EN	PMIC On Request Enable. The value written to PK_EN will be asserted on output signal snvs_lp_pk_en. That signal is used to turn off the pullup/pulldown circuitry in the PMIC I/O pad.
21–20 ON_TIME	The ON_TIME field is used to configure the period of time after BTN is asserted before pmic_en_b is asserted to turn on the SoC power. 00: 500msec off->on transition time 01: 50msec off->on transition time 10: 100msec off->on transition time 11: 0msec off->on transition time

Table continues on the next page...

## SNVS\_LPCR field descriptions (continued)

Field	Description
19–18 DEBOUNCE	This field configures the amount of debounce time for the BTN input signal. 00: 50msec debounce 01: 100msec debounce 10: 500msec debounce 11: 0msec debounce
17–16 BTN_PRESS_ TIME	Button press time out values for PMIC Logic. 00 : 5 secs 01 : 10 secs 10 : 15 secs 11 : long press disabled (pmic_en_b will not be asserted regardless of how long BTN is asserted)
15 -	This field is reserved.
14–10 LPCALB_VAL	LP Calibration Value Defines signed calibration value for SRTC. This field can be programmed only when SRTC calibration is disabled and not locked, i.e. when LPCALB_EN, LPCALB_SL, and LPCALB_HL bits are not set. This is a 5-bit 2's complement value. Hence, the allowable calibration values are in the range from -16 to +15 counts per 32768 ticks of the counter clock  00000 +0 counts per each 32768 ticks of the counter clock 00001 +1 counts per each 32768 ticks of the counter clock 00010 +2 counts per each 32768 ticks of the counter clock 01111 +15 counts per each 32768 ticks of the counter clock 10000 -16 counts per each 32768 ticks of the counter clock 10001 -15 counts per each 32768 ticks of the counter clock 11110 -2 counts per each 32768 ticks of the counter clock 11111 -1 counts per each 32768 ticks of the counter clock
9 -	This field is reserved.
8 LPCALB_EN	LP Calibration Enable When set, enables the SRTC calibration mechanism. This bit cannot be changed once LPCALB_SL or LPCALB_HL bit is set.  0 SRTC Time calibration is disabled. 1 SRTC Time calibration is enabled.
7 PWR_GLITCH_ EN	By default the detection of a power glitch does not cause the pmic_en_b signal to be asserted. Setting the Power Glitch Enable bit to 1 enables the power glitch tamper event for the PMIC.  0 - disabled 1 - enabled
6 TOP	Turn off System Power Asserting this bit causes a signal to be sent to the Power Management IC to turn off the system power. This bit will clear once power is off. This bit is only valid when the Dumb PMIC is enabled.  0 Leave system power on. 1 Turn off system power.

Table continues on the next page...

**SNVS\_LPCR field descriptions (continued)**

<b>Field</b>	<b>Description</b>
5 DP_EN	<p>Dumb PMIC Enabled</p> <p>When set, software can control the system power. When cleared, the system requires a Smart PMIC to automatically turn power off.</p> <p>0 Smart PMIC enabled. 1 Dumb PMIC enabled.</p>
4 SRTC_INV_EN	<p>Secure Real Time Counter Invalidation Enable</p> <p>When set, the SRTC is invalidated (SRTC_ENV bit is cleared) in the case of security violation. This field cannot be changed once SRTC_SL or SRTC_HL bit is set.</p> <p>0 SRTC stays valid in the case of security violation. 1 SRTC is invalidated in the case of security violation.</p>
3 LPWUI_EN	<p>LP Wake-Up Interrupt Enable</p> <p>This interrupt line should be connected to the external pin and is intended to inform the external chip about an SNVS_LP event (tamper event, MC rollover, SRTC rollover, or time alarm ). This wake-up signal can be asserted only when the chip (HP section) is powered down, and the LP section is isolated.</p> <p>0 LP wake-up interrupt is disabled. 1 LP wake-up interrupt is enabled.</p>
2 MC_ENV	<p>Monotonic Counter Enable and Valid</p> <p>When set, the MC can be incremented (by write transaction to the LPSMCMR or LPSMCLR). This bit cannot be changed once MC_SL or MC_HL bit is set.</p> <p>0 MC is disabled or invalid. 1 MC is enabled and valid.</p>
1 LPTA_EN	<p>LP Time Alarm Enable</p> <p>When set, the SNVS functional interrupt is asserted if the LP Time Alarm Register is equal to the 32 MSBs of the secure real time counter.</p> <p>0 LP time alarm interrupt is disabled. 1 LP time alarm interrupt is enabled.</p>
0 SRTC_ENV	<p>Secure Real Time Counter Enable and Valid</p> <p>When set, the SRTC becomes operational. This bit cannot be changed once SRTC_SL or SRTC_HL bit is set.</p> <p>0 SRTC is disabled or invalid. 1 SRTC is enabled and valid.</p>

## 44.9.11 SNVS\_LP Status Register (SNVS\_LPSR)

The SNVS\_LP Status Register reflects the internal state and behavior of the SNVS\_LP.

Address: 20C\_C000h base + 4Ch offset = 20C\_C04Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	Security	Security	Security											Reserved	SPO	EO	Security
W															w1c	w1c	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					Security	Security	Security	Security	Security	Security	Security	Security	MCR	SRTC	LPTA
W														w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0

## SNVS\_LPSR field descriptions

Field	Description
31 Security	Security-related field.
30 Security	Security-related field.
29–20 Security	Security-related field.
19 -	This field is reserved.
18 SPO	Set Power Off The SPO bit is set when the set_pwr_off_irq interrupt is triggered, which happens when software writes a 1 to the TOP bit in the LPCR or when the power button is pressed longer than the configured debounce time. Writing to the SPO bit will clear the set_pwr_off_irq interrupt.  0 Emergency Off was not detected. 1 Emergency Off was detected..
17 EO	Emergency Off This bit is set when a power off is requested.  0 Emergency off was not detected. 1 Emergency off was detected.
16 Security	Security-related field.
15–11 -	This field is reserved.
10 Security	Security-related field.
9 Security	Security-related field.
8 Security	Security-related field.
7 Security	Security-related field.
6 Security	Security-related field.
5 Security	Security-related field.
4 Security	Security-related field.
3 Security	Security-related field.
2 MCR	Monotonic Counter Rollover.  0 MC has not reached its maximum value. 1 MC has reached its maximum value.
1 SRTCR	Secure Real Time Counter Rollover.

*Table continues on the next page...*

**SNVS\_LPSR field descriptions (continued)**

Field	Description
	0 SRTC has not reached its maximum value. 1 SRTC has reached its maximum value.
0 LPTA	LP Time Alarm.  0 No time alarm interrupt occurred. 1 A time alarm interrupt occurred.

**44.9.12 SNVS\_LP Secure Real Time Counter MSB Register (SNVS\_LPSRTCMR)**

The SNVS\_LP Secure Real Time Counter MSB register contains the most-significant bits of the secure real time counter.

Address: 20C\_C000h base + 50h offset = 20C\_C050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																SRTC															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SNVS\_LPSRTCMR field descriptions**

Field	Description
31–15 -	This field is reserved.
14–0 SRTC	LP Secure Real Time Counter most significant 32 bits  This register can be programmed only when SRTC is not active and not locked, meaning the SRTC_ENV, SRTC_SL, and SRTC_HL bits are not set.

**44.9.13 SNVS\_LP Secure Real Time Counter LSB Register (SNVS\_LPSRTCLR)**

The SNVS\_LP Secure Real Time Counter LSB register contains the 32 least-significant bits of the secure real time counter.

Address: 20C\_C000h base + 54h offset = 20C\_C054h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	



**SNVS\_LPSRTCLR field descriptions**

Field	Description
31–0 SRTC	LP Secure Real Time Counter least significant 32 bits  This register can be programmed only when SRTC is not active and not locked, meaning the SRTC_ENV, SRTC_SL, and SRTC_HL bits are not set.

**44.9.14 SNVS\_LP Time Alarm Register (SNVS\_LPTAR)**

The SNVS\_LP Time Alarm register contains the 32-bit LP Time Alarm value.

Address: 20C\_C000h base + 58h offset = 20C\_C058h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	<div>LPTA</div>																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SNVS\_LPTAR field descriptions**

Field	Description
31–0 LPTA	LP Time Alarm  This register can be programmed only when the LP time alarm is disabled (LPTA_EN bit is not set).

**44.9.15 SNVS\_LP Secure Monotonic Counter MSB Register (SNVS\_LPSMCMR)**

The SNVS\_LP Secure Monotonic Counter MSB Register contains the monotonic counter era bits and the most significant 16 bits of the monotonic counter. The monotonic counter is incremented by one if there is a write command to the LPSMCMR or LPSMCLR register.

Address: 20C\_C000h base + 5Ch offset = 20C\_C05Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MC_ERA_BITS																MON_COUNTER															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SNVS\_LPSMCMR field descriptions**

Field	Description
31–16 MC_ERA_BITS	Monotonic Counter Era Bits

*Table continues on the next page...*

**SNVS\_LPSMCMR field descriptions (continued)**

Field	Description
	These bits are inputs to the module and typically connect to fuses.
15–0 MON_COUNTER	Monotonic Counter Most Significant 16 Bits The MC is incremented by one when: <ul style="list-style-type: none"> <li>• A write transaction to the LPSMCMR or LPSMCLR register is detected.</li> <li>• The MC_ENV bit is set.</li> <li>• MC_SL and MC_HL bits are not set.</li> </ul>

**44.9.16 SNVS\_LP Secure Monotonic Counter LSB Register (SNVS\_LPSMCLR)**

The SNVS\_LP Secure Monotonic Counter LSB Register contains the 32 least significant bits of the monotonic counter. The MC is incremented by one if there is a write command to the LPSMCMR or LPSMCLR register.

Address: 20C\_C000h base + 60h offset = 20C\_C060h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
	MON_COUNTER																															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SNVS\_LPSMCLR field descriptions**

Field	Description
31–0 MON_COUNTER	Monotonic Counter bits The MC is incremented by one when: <ul style="list-style-type: none"> <li>• A write transaction to the LPSMCMR or LPSMCLR Register is detected.</li> <li>• The MC_ENV bit is set.</li> <li>• MC_SL and MC_HL bits are not set.</li> </ul>

**44.9.17 SNVS\_LP General Purpose Register (SNVS\_LPGPR)**

The SNVS\_LP General Purpose Register provides a 32 bit read write register, which can be used by any application for retaining 32 bit data during a power-down mode.

Address: 20C\_C000h base + 68h offset = 20C\_C068h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	<div>GPR</div>																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SNVS\_LPGPR field descriptions**

Field	Description
31–0 GPR	General Purpose Register When GPR_SL or GPR_HL bit is set, the register cannot be programmed.

**44.9.18 SNVS\_HP Version ID Register 1 (SNVS\_HPVIDR1)**

The SNVS\_HP Version ID Register 1 is a read-only register that contains the current version of the SNVS. The version consists of a module ID, a major version number, and a minor version number.

Address: 20C\_C000h base + BF8h offset = 20C\_CBF8h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IP_ID																MAJOR_REV								MINOR_REV							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

**SNVS\_HPVIDR1 field descriptions**

Field	Description
31–16 IP_ID	SNVS block ID
15–8 MAJOR_REV	SNVS block major version number
7–0 MINOR_REV	SNVS block minor version number

**44.9.19 SNVS\_HP Version ID Register 2 (SNVS\_HPVIDR2)**

The SNVS\_HP Version ID Register 2 is a read-only register that indicates the current version of the SNVS. The version consists of the following fields: integration options, ECO revision, and configuration options.

Address: 20C\_C000h base + BFCh offset = 20C\_CBFCh

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	IP_ERA								INTG_OPT								ECO_REV								CONFIG_OPT							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SNVS\_HPVIDR2 field descriptions**

Field	Description
31–24 IP_ERA	Era of the IP design 00h - Era 1 or 2 03h - Era 3
23–16 INTG_OPT	SNVS Integration Option
15–8 ECO_REV	SNVS ECO Revision
7–0 CONFIG_OPT	SNVS Configuration Option

# Chapter 45

## Shared Peripheral Bus Arbiter (SPBA)

### 45.1 Overview

The Shared Peripheral Bus Arbiter (SPBA) is a three-to-one IP Bus interface arbiter. Three masters arbitrate for shared peripheral access through the SPBA.

The SPBA has three primary functions:

- The IP Bus Line switches a master to one peripheral
- The Masters arbiter arbitrates between the three masters to solve concurrent access or restricted access to peripherals
- The Control Registers and Ownership Control includes a set of registers which are reachable through software and permit the access scheme to be defined for each peripheral (Resource Ownership and Access Control). It generates signals for the external steering logic of interrupts and DMA signals.

The figure below shows the SPBA block diagram

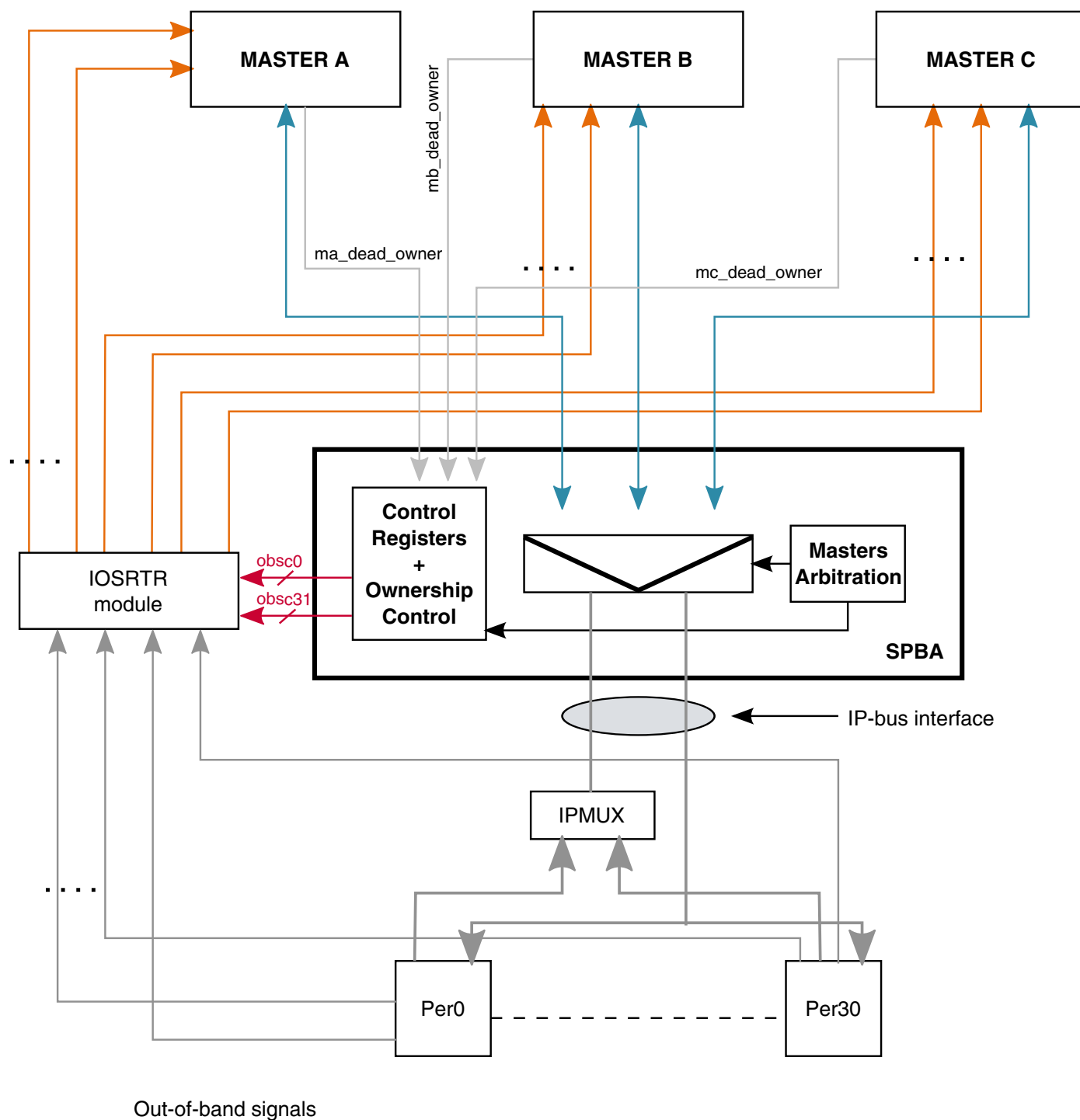


Figure 45-1. SPBA Block Diagram

### 45.1.1 Features

The SPBA includes the following features:

- Three IP Bus masters arbitration: Master A, B and C
- Support for DMA masters
- 32-bit data
- Supports up to 31 shared peripherals, each consuming 16 kilobytes of address space
- SPBA can be considered the 32nd peripheral, used for resource ownership and access control of the 31 peripherals
- Provides 31 sets of out of band steering control (OBSC) signals to the off-block steering logic
- Operating frequency up to 67 MHz
- Clocks: ipg\_clk, ipg\_clk\_s

### 45.1.2 Modes of operation

SPBA behavior is transparent when accessing a peripheral, though it has these distinct modes of operation.

#### Reset/Abort

The SPBA has a hardware reset which initializes all registers, arbitration and peripherals rights registers (PRRs).

An abort signal input is provided allowing each master to abort its current access and release ownership (in case of master reset sequence).

#### Functional

Once a master request is granted, its IP Bus signals are steered to the requested peripheral.

#### Standby

No clock needed. The SPBA needs clocks only during access to the PRRs, arbitration, and abort phases. It generates two clock enable signals indicating when the clocks must be provided.

#### Configuration

During this phase, a master accesses the SPBA PRRs. The SPBA memory-mapped registers are seen as a shared peripheral.

## 45.2 Clocks

The table found here describes the clock sources for SPBA.

Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 45-1. SPBA Clocks**

Clock name	Clock Root	Description
ipg_clk	ipg_clk_root	Peripheral clock
ipg_clk_s	ipg_clk_root	Peripheral access clock

## 45.3 Functional description

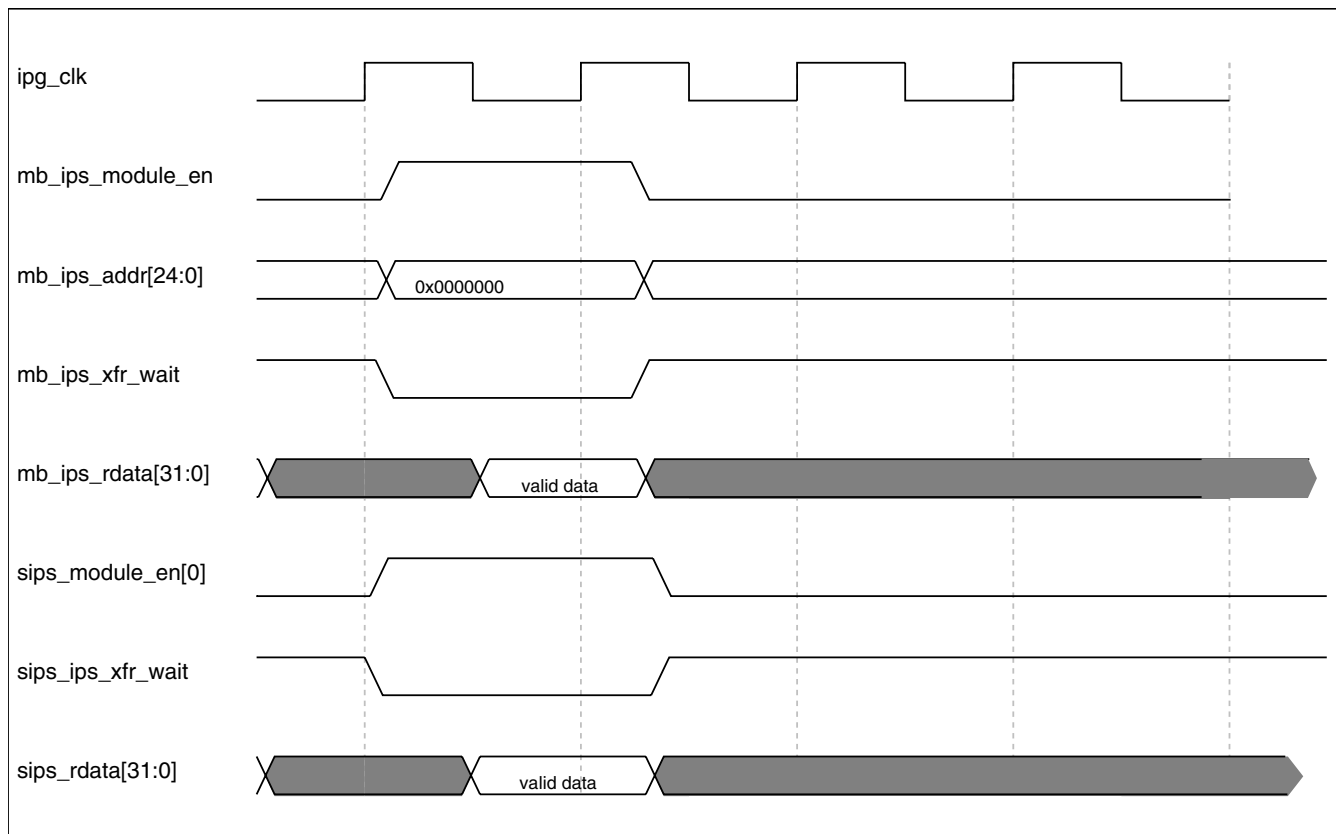
### 45.3.1 Masters arbitration

The arbitration mechanism determines which port will control the master port, based on a simple round-robin arbitration scheme.

There are several use cases to consider.

- Only one master request per access. The master is switched to the shared peripheral bus, without arbitration. [Figure 45-2](#) shows the MB request on the global module enable signal, served without wait state.
- If two masters simultaneously access SPBA, the last granted master is held off using the <master>\_ips\_xfr\_wait output signal (default value is high). When the master is granted sips\_xfr\_wait, shared IP Bus peripheral is connected to <master>\_ips\_xfr\_wait outputs.
- If three masters simultaneously access SPBA, then the last two granted masters are held off using <master>\_ips\_xfr\_wait. [Figure 45-3](#) shows a case in which the last two accesses granted are MA and MB. The requests are used even if they are in the same cycle.
- If after reset, at the first multiple access, no master has been granted, the priority is static: Master A (MA), Master B (MB) and last Master C (MC) port.
- No master request. No master switch to shared peripherals.

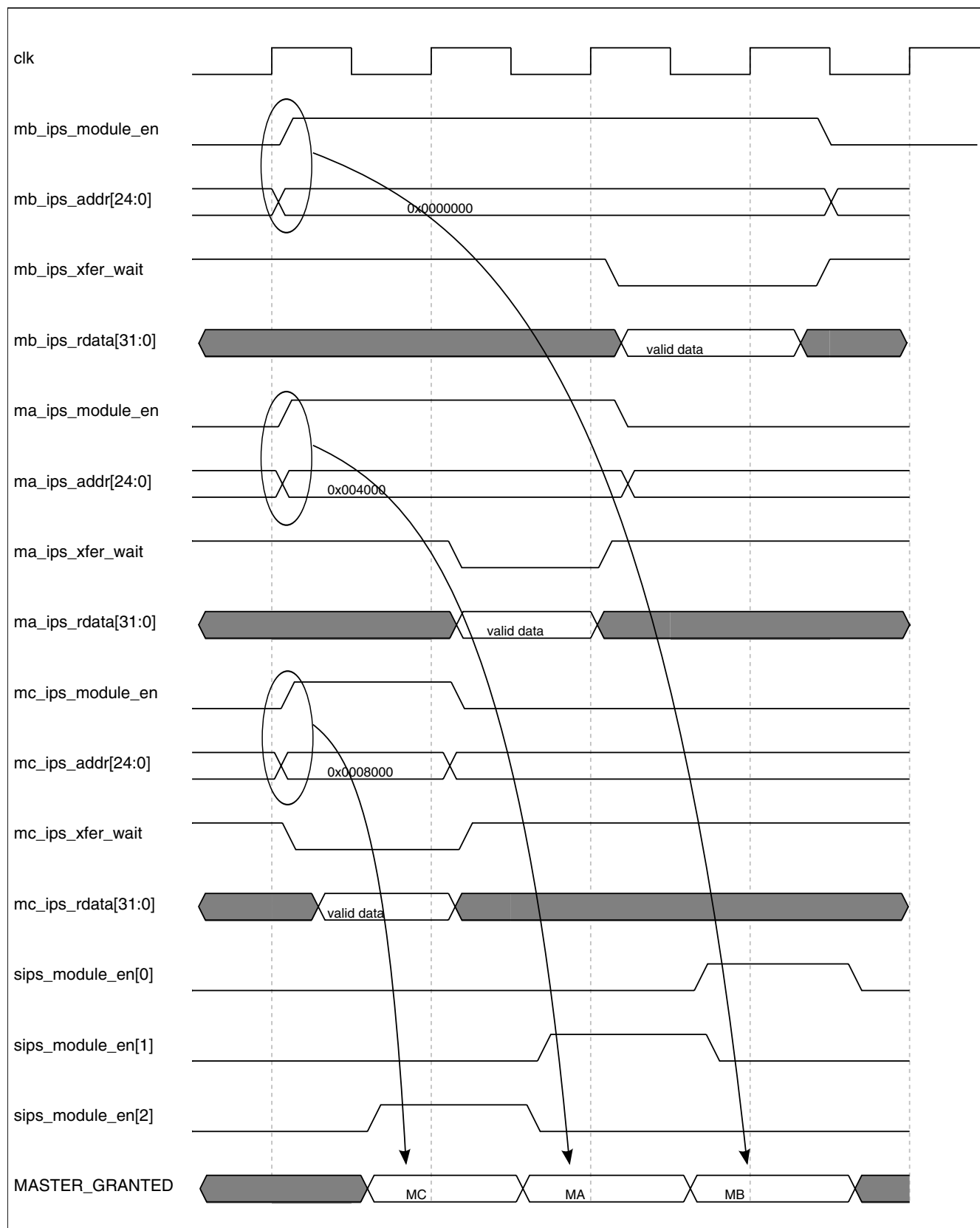




**Figure 45-2. Example of one master request, no SPBA arbitration**

The following figure assumes MA and MB have been the last two masters granted in the previous transfers (MA then MB).

## Functional description



**Figure 45-3. Example of three master requests: Masters already granted are "waited";**

## **45.4 Resource ownership control**

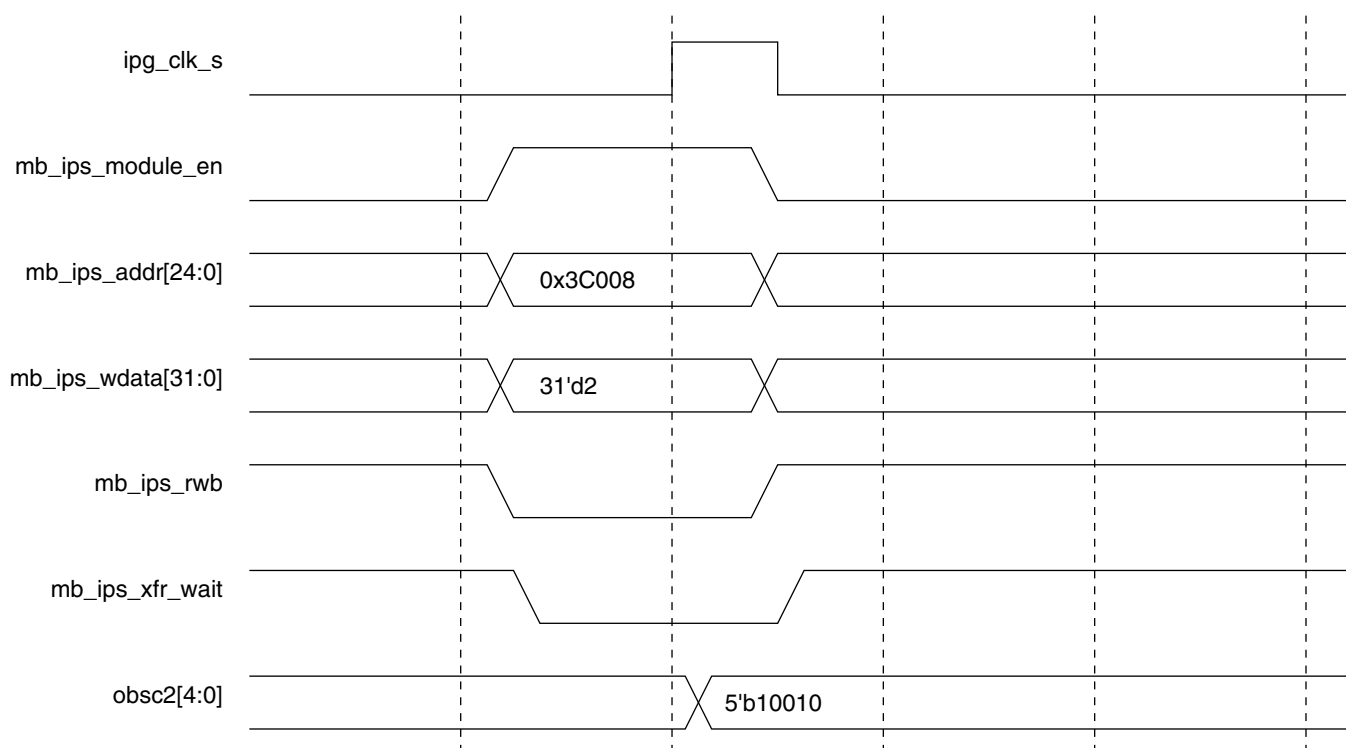
The resource ownership control regulates access to the shared peripherals and determines the steering of out-of-band signals.

### **45.4.1 Access control**

### 45.4.1.1 Peripheral access

The peripheral access (resource access) of the requesting master is given by the corresponding RAR bit of the Peripheral Right Register. It determines if the master has access privilege to the resource.

Any attempt at access made by a requesting master whose access privilege bit is not set (in the PRR) is terminated with a bus error (<master>\_ips\_xfr\_err is asserted by SPBA logic). The master that owns the resource can lock the peripheral for itself and/or grant other masters access to the peripheral by setting the appropriate bit(s) in the RAR field.



Master B is taking ownership of peripheral 2 by writing 3'b010 in the SPBA peripheral 2 right register (rarfield)  
 This ownership can be checked on obsc2 output as roi2[1:0] = 2'b10 and rar2[2:0] = 3'b010  
 (obsc[4:0] = {roi2[1], roi2[0], rar2[2], rar2[1], rar2[0]})

**Figure 45-4. Example of one master B gaining ownership of peripheral 2**

### 45.4.1.2 Peripheral Right Register access

The ROI bits of the Peripheral Right Register (PRR) determine which master is allowed to make write access to PRR. The identification of the requesting master is compared to the ROI bits of the PRR to determine if the master has ownership of the corresponding register.

Any attempted write access to a PRR already owned by another master will be ignored.

### 45.4.2 Owner election

When the peripheral is not owned by any master (ROI="00", after coming out of reset for instance), the first master to perform successfully a write to the RAR bits of the PRR is granted ownership of the peripheral and its associated PRR.

After writing to the PRR (RAR bit(s)), the master must read it back to make sure that it was granted ownership. If the RMO field is 2'b11, then the ownership claim is successful. If RMO is 2'b10, another master claimed ownership before this master was able to complete its write. This resolves the case in which two or more masters attempt to write the PRR at the same time; only the first master will be granted ownership. However all masters must read the PRR to determine if this case occurred, and if so, whether they were the first master which was granted ownership.

#### NOTE

A master that has been granted ownership of the PRR does not automatically have the right access to the peripheral; it must still set its own RAR bits in the PRR to access the peripheral.

### 45.4.3 Ending ownership

Ownership may be voluntarily ended by the owning master, or automatically upon assertion of a master-specific dead\_owner signal.

The former is appropriate for software-controlled yielding of ownership. The latter is appropriate for automatic yielding of ownership when the owner has gone into reset.

When a master is reset, it clears the ROI bits of the PRRs owned by the corresponding master. When the owner is dead (in reset), all peripherals previously owned by that master must be changed to the un-owned state.

#### NOTE

It is the programmer's responsibility to make sure the peripherals are placed in an appropriate state before ending ownership.

### 45.4.3.1 Software Controlled Ownership Ending

The ROI bits will be automatically cleared when the master that owns the PRR access right clears (write) the RAR bits ([Table 2](#)).

It will then end the ownership of the PRR.

### 45.4.4 The Un-owned State

During the time when the peripheral is un-owned (i.e the ROI field contains all 0's), all masters have full access to it (RAR bits can then be modified by a master if ROI[1:0] = 2'b0).

In such cases it is necessary for software to ensure any necessary coherency in the resource, there is no hardware protection.

## 45.5 SPBA Memory Map/Register Definition

The SPBA control registers (Peripheral Right Registers) are mapped as a virtual shared peripheral.

SPBA can support up to 31 shared peripherals. Each of them has its own Peripheral Right Register (PRR) accessible within the SPBA memory-mapped registers, and consists of the Requesting Master Owner, the Resource Owner ID and the Resource Access Right fields.

**SPBA memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
203_C000	Peripheral Rights Register (SPBA_PRR0)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C004	Peripheral Rights Register (SPBA_PRR1)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C008	Peripheral Rights Register (SPBA_PRR2)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C00C	Peripheral Rights Register (SPBA_PRR3)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C010	Peripheral Rights Register (SPBA_PRR4)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C014	Peripheral Rights Register (SPBA_PRR5)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C018	Peripheral Rights Register (SPBA_PRR6)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C01C	Peripheral Rights Register (SPBA_PRR7)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C020	Peripheral Rights Register (SPBA_PRR8)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>

*Table continues on the next page...*

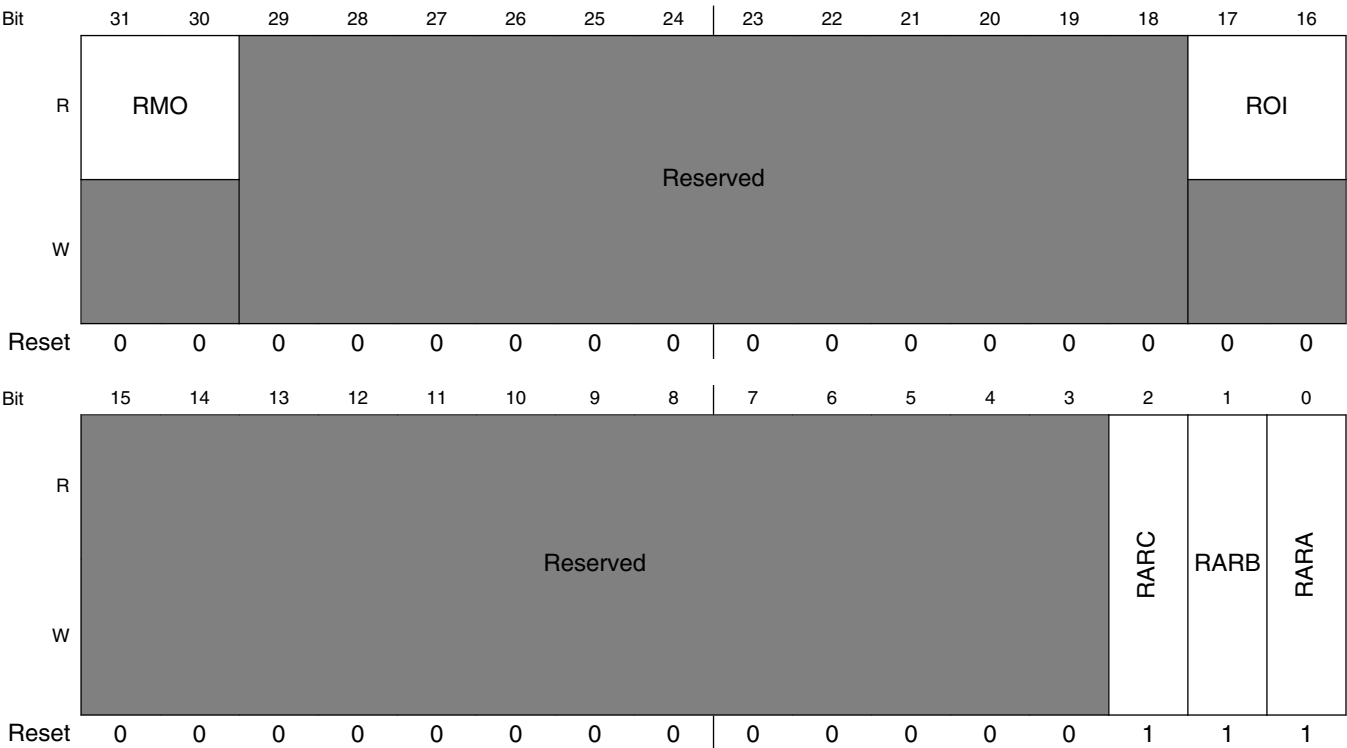
## SPBA memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
203_C024	Peripheral Rights Register (SPBA_PRR9)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C028	Peripheral Rights Register (SPBA_PRR10)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C02C	Peripheral Rights Register (SPBA_PRR11)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C030	Peripheral Rights Register (SPBA_PRR12)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C034	Peripheral Rights Register (SPBA_PRR13)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C038	Peripheral Rights Register (SPBA_PRR14)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C03C	Peripheral Rights Register (SPBA_PRR15)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C040	Peripheral Rights Register (SPBA_PRR16)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C044	Peripheral Rights Register (SPBA_PRR17)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C048	Peripheral Rights Register (SPBA_PRR18)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C04C	Peripheral Rights Register (SPBA_PRR19)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C050	Peripheral Rights Register (SPBA_PRR20)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C054	Peripheral Rights Register (SPBA_PRR21)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C058	Peripheral Rights Register (SPBA_PRR22)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C05C	Peripheral Rights Register (SPBA_PRR23)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C060	Peripheral Rights Register (SPBA_PRR24)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C064	Peripheral Rights Register (SPBA_PRR25)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C068	Peripheral Rights Register (SPBA_PRR26)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C06C	Peripheral Rights Register (SPBA_PRR27)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C070	Peripheral Rights Register (SPBA_PRR28)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C074	Peripheral Rights Register (SPBA_PRR29)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C078	Peripheral Rights Register (SPBA_PRR30)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>
203_C07C	Peripheral Rights Register (SPBA_PRR31)	32	R/W	0000_0007h	<a href="#">45.5.1/2772</a>

### 45.5.1 Peripheral Rights Register (SPBA\_PRRn)

This register controls master ownership and access for a peripheral.

Address: 203\_C000h base + 0h offset + (4d × i), where i=0d to 31d



SPBA\_PRRn field descriptions

Field	Description
31–30 RMO	Requesting Master Owner. This 2-bit register field indicates if the corresponding resource is owned by the requesting master or not. This register is reset to 2'b0 if ROI = 2'b0.  00 <b>UNOWNED</b> — The resource is unowned. 01 Reserved. 10 <b>ANOTHER_MASTER</b> — The resource is owned by another master. 11 <b>REQUESTING_MASTER</b> — The resource is owned by the requesting master.
29–18 -	This field is reserved. Reserved
17–16 ROI	Resource Owner ID. This field indicates which master (one at a time) can access to the PRR for rights modification. This is a read-only register.  After reset, ROI bits are cleared ("00" -> un-owned resource).

Table continues on the next page...



## SPBA\_PRRn field descriptions (continued)

Field	Description
	<p>A master performing a write access to the an un-owned PRR will get its ID automatically written into ROI, while modifying RARx bits. It can then read back the RMO, RAR, ROI bits to make sure RMO returns the right value, ROI bits contain its ID and RARx bits are correctly asserted. Then no other master (whom ID is different from the one stored in ROI) will be able to modify RAR fields.</p> <p>Owner master of a peripheral can assert its dead_owner signal, or write 1'b0 in the RARx to release the ownership (ROI[1:0] reset to 2'b0).</p> <p>00 <b>UNOWNED</b> — Unowned resource.</p> <p>01 <b>MASTER_A</b> — The resource is owned by master A port.</p> <p>10 <b>MASTER_B</b> — The resource is owned by master B port.</p> <p>11 <b>MASTER_C</b> — The resource is owned by master C port.</p>
15–3 -	<p>This field is reserved.</p> <p>Reserved</p>
2 RARC	<p>Resource Access Right. Control and Status bit for master C.</p> <p>This field indicates whether master C can access the peripheral. From 0 up to 3 masters can have permission to access a resource (all the master can be granted on a peripheral, but only one access at a time will be granted by SPBA).</p> <p>0 <b>PROHIBITED</b> — Access to peripheral is not allowed.</p> <p>1 <b>ALLOWED</b> — Access to peripheral is granted.</p>
1 RARB	<p>Resource Access Right. Control and Status bit for master B.</p> <p>This field indicates whether master B can access the peripheral. From 0 up to 3 masters can have permission to access a resource (all the master can be granted on a peripheral, but only one access at a time will be granted by SPBA).</p> <p>0 <b>PROHIBITED</b> — Access to peripheral is not allowed.</p> <p>1 <b>ALLOWED</b> — Access to peripheral is granted.</p>
0 RARA	<p>Resource Access Right. Control and Status bit for master A.</p> <p>This field indicates whether master A can access the peripheral. From 0 up to 3 masters can have permission to access a resource (all the master can be granted on a peripheral, but only one access at a time will be granted by SPBA).</p> <p>0 <b>PROHIBITED</b> — Access to peripheral is not allowed.</p> <p>1 <b>ALLOWED</b> — Access to peripheral is granted.</p>



## Chapter 46

# Sony/Philips Digital Interface (SPDIF)

### 46.1 Introduction

The Sony/Philips Digital Interface (SPDIF) audio block is a stereo transceiver that allows the processor to receive and transmit digital audio.

The SPDIF transceiver allows the handling of both SPDIF channel status (CS) and User (U) data and includes a frequency measurement block that allows the precise measurement of an incoming sampling frequency.

A recovered clock is provided to drive both internal and external components in the system such as ESAI ports, as well as external A/Ds or D/As, with clocking control provided via related registers.

As the SPDIF internal data width is 24-bit, the eight most-significant bits of all registers return zeros.

The figure below shows a block diagram of the SPDIF transceiver data paths (receiver and transmitter) and its interface.

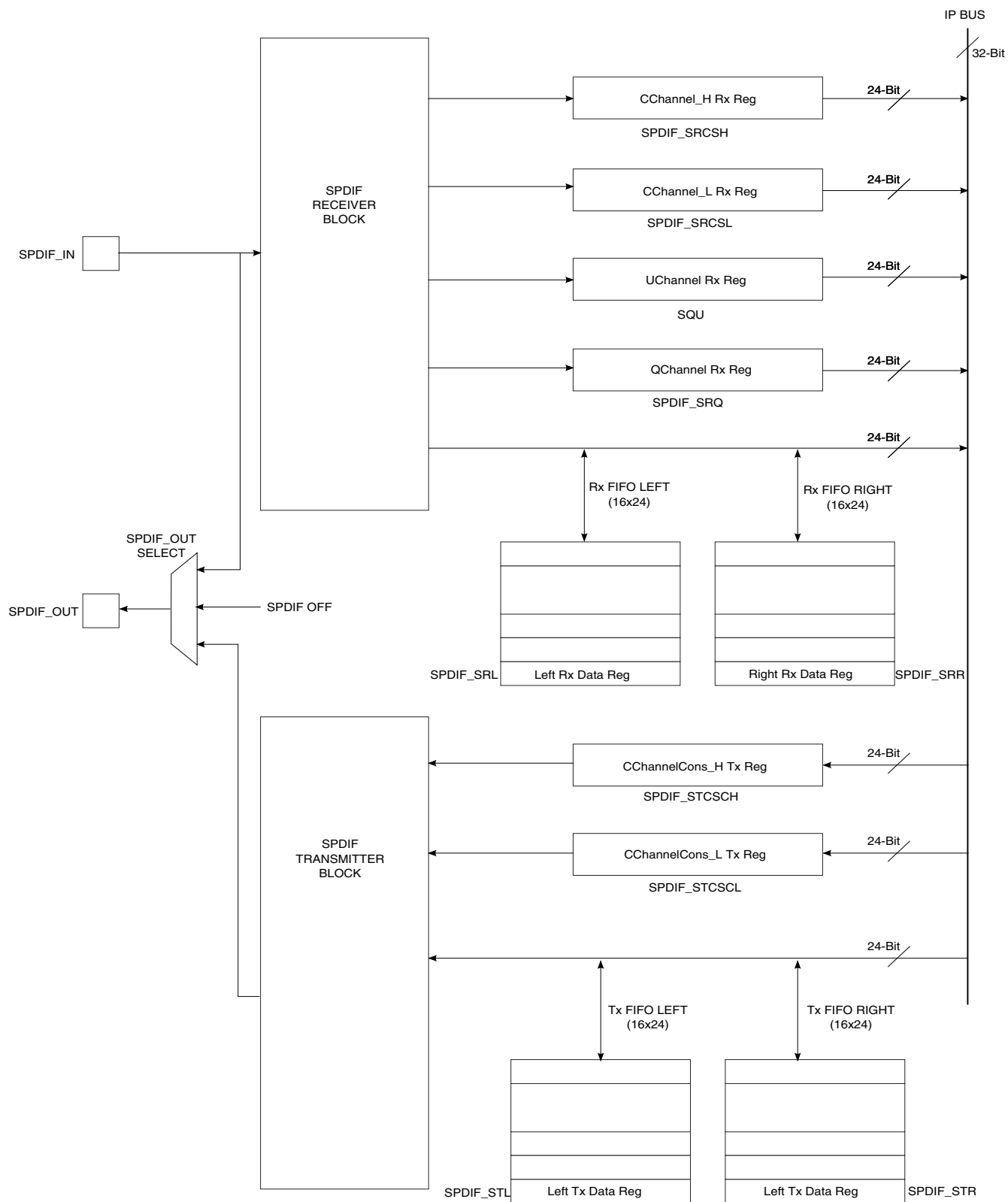


Figure 46-1. SPDIF Transceiver Data Interface Block Diagram

### 46.1.1 Overview

The SPDIF is composed of two parts: SPDIF Receiver and SPDIF Transmitter.

The SPDIF receiver extracts the audio data from each SPDIF frame and places the data in the SPDIF Rx left and right FIFOs. The Channel Status and User Bits are also extracted from each frame and placed in the corresponding registers. The SPDIF receiver also provides a bypass option for direct transfer of the SPDIF input signal to the SPDIF transmitter.

For the SPDIF transmitter, the audio data is provided by the processor via the SPDIFTxLeft and SPDIFTxRight registers. The Channel Status bits are also provided via the corresponding registers. The SPDIF transmitter generates a SPDIF output bitstream in the biphas mark format (IEC60958), which consists of audio data, channel status and user bits.

In the SPDIF transmitter, the IEC60958 biphas bit stream is generated on both edges of the SPDIF Transmit clock. The SPDIF Transmit clock is generated by the SPDIF internal clock generate block and the sources are from outside of the SPDIF block. For the SPDIF receiver, it can recover the SPDIF Rx clock. [Figure 46-2](#) shows the clock structure of the SPDIF transceiver.

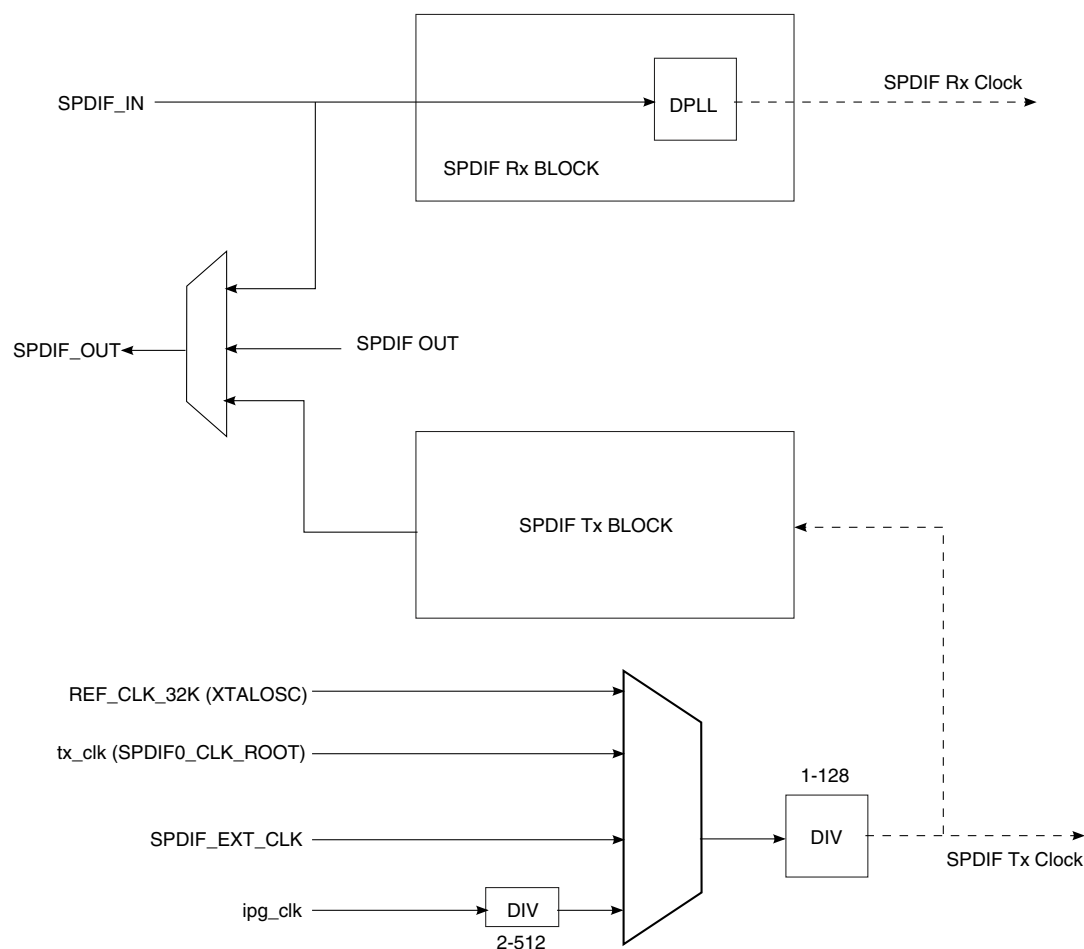


Figure 46-2. SPDIF Transceiver Clock Diagram

# 46.2 External Signals

Table 46-1. SPDIF External Signals

Signal	Description	Pad	Mode	Direction
SPDIF_EXT_CLK	External clock signal	AUD_MCLK	ALT6	I
		ECSPi2_SCLK	ALT1	
		FEC_REF_CLK	ALT6	
SPDIF_IN	Input line signal	FEC_TX_EN	ALT2	I
		I2C2_SCL	ALT2	
		SD2_DAT5	ALT4	

Table continues on the next page...

**Table 46-1. SPDIF External Signals  
(continued)**

Signal	Description	Pad	Mode	Direction
SPDIF_OUT	Output line signal	FEC_TXD1	ALT2	O
		I2C2_SDA	ALT2	
		SD2_DAT4	ALT4	
		SD2_RST	ALT3	

## 46.3 Clocks

The table found here describes the clock sources for SPDIF.

Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 46-2. SPDIF Clocks**

Clock name	Clock Root	Description
gclkw_t0	ipg_clk_root	Global clock
ipg_clk_s	ipg_clk_root	Peripheral access clock
tx_clk	spdif0_clk_root	Module Tx clock

## 46.4 Functional Description

### 46.4.1 SPDIF Receiver

The SPDIF receiver extracts the audio data from each SPDIF frame and places the data in Rx left and right FIFOs.

The Tx left and right FIFOs are 16-deep and 24-bit-wide (equal to the audio data width). The Channel Status and User Bits are also extracted from each frame and placed in corresponding registers. The SPDIF receiver also provides a bypass option for direct transfer of the SPDIF input signal to the SPDIF transmitter.

The SPDIF receiver handles the main data audio stream and recovers the bit clock from the SPDIF input signal. The sample rate can be determined from the frequency measuring block. Additionally, the receiver supports the SPDIF C and U channels. The SPDIF C and U channel data is interfaced directly to memory-mapped registers.

All the data registers are controlled by the Interrupt Control Block and transferred to the memory-mapped IP bus.

The following functions are performed by the SPDIF receiver:

- Audio Data Reception see [Audio Data Reception](#)
- Channel Status bits Reception see [Channel Status Reception](#)
- U Channel bits Reception see [User Bit Reception](#)
- Validity Flag Reception see [Validity Flag Reception](#)
- SPDIF Receiver Exception support see [SPDIF Receiver](#)
- SPDIF Lock Detection

#### **46.4.1.1 Audio Data Reception**

The SPDIF Receiver block extracts the audio data from the IEC60958 stream, and outputs this via Rx left and right FIFOs to the memory-mapped registers SPDIFRxLeft and SPDIFRxRight.

Data from the SPDIF receiver is buffered in receive FIFO, and can be read by the processor from the memory-mapped registers.

##### **• SPDIF receiver data registers - Behavior on overrun, underrun**

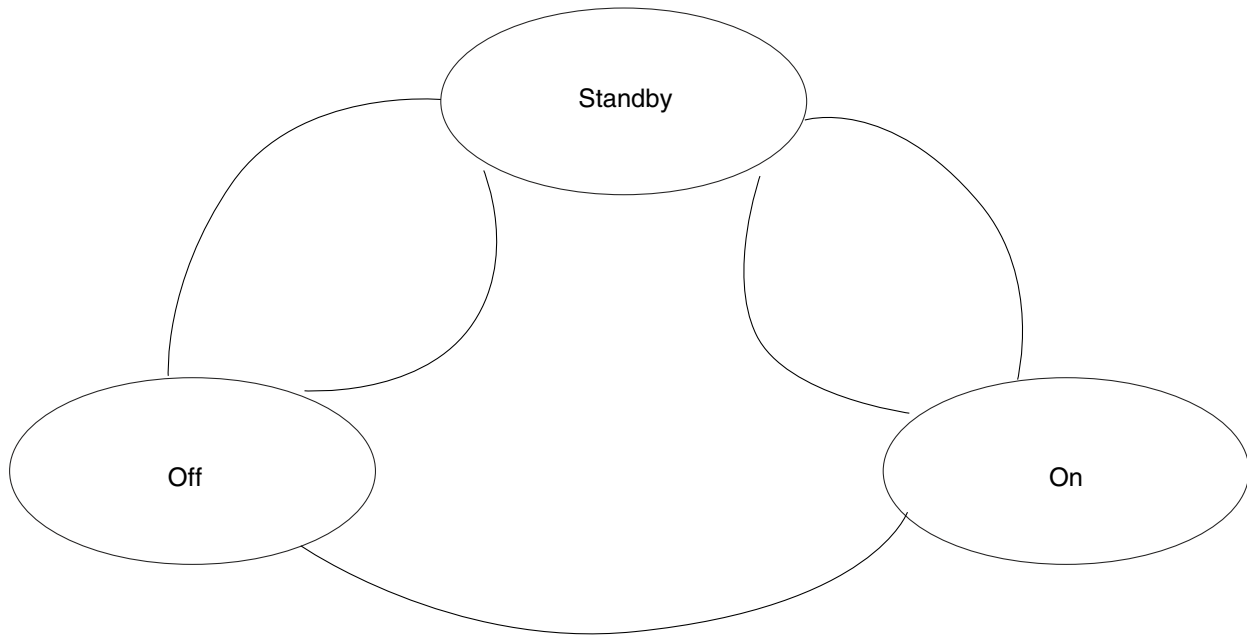
The SPDIF Data Receive registers (SPDIFRxLeft and SPDIFRxRight) have individual FIFOs for left and right channel. As a result, there is always the possibility that left and right FIFOs may go out of sync due to FIFO underruns and FIFO overruns that affect only one part (left or right) of any FIFOs. To prevent this from happening, hardware has been added to the device. Two mechanisms to prevent mismatch between the FIFOs are available.

If a SPDIF Data Rx FIFO overrun occurs on e.g. the right half of the FIFO, the sample that caused the overrun is not written to the right half (due to overrun). Special hardware will make sure the next sample is not written to the left half of the FIFO. If the overrun occurs on the left half of the FIFO, the next sample is not written to the right half of the FIFO.

##### **• SPDIF receiver data registers - Automatic resynchronization of FIFOs**

An automatic FIFO resynchronization feature is available. It can be enabled and disabled separately for every FIFO. If it is enabled, the hardware will check to see if the left and right FIFOs are in sync. If that is not the case, it will set the filling pointer of the right FIFO to be equal to the filling pointer of the left FIFO.





**Figure 46-3. FIFO Auto-resync Controller State Machine**

The operation is explained from the state diagram shown above. Every FIFO auto-resync controller has a state machine with 3 states: Off, StandBy and On. In the On state, the filling of the left FIFO is compared with the filling of right, and if they are not equal, right is made equal to left, and an interrupt is generated.

The controller will stay in Off state when the feature is disabled. When not disabled, the state machine will go to Off state on any processor read or write to the FIFO. It will go from On or Off to Standby on any left sample read from SPDIF Tx FIFOs, or on any left sample write to SPDIF Rx FIFOs. The controller will go from Standby to On on any right sample read from SPDIF Tx FIFO, or on any right sample write to SPDIF Rx FIFO. There is a control bit in the SPDIFConfig register to enable/disable the feature for the SPDIF Rx FIFO and SPDIF Tx FIFO.

#### 46.4.1.1.1 Application Note

The automatic FIFO resynchronization can be switched on, and will avoid all mismatches between left and right FIFOs, if the software obeys the following rules: 1. When the left data is read or written to the left FIFO, in the same place of the program, data must be read or written to the right FIFO. Maximum time difference between left and right is 1/2 sample clock. (E.g. if sample frequency is 44 KHz, approximately 10 micro-seconds. For 88 KHz, approximately 5 micro-seconds.) 2. Write/read data to FIFO s at least 2 samples

at the time. If there is a mismatch Left-Right, the resync logic may go on only 1 sample clock after last data is read/written to the FIFO. Also acceptable is polling the FIFO, if at least part of the time 2 samples will be read/written to it.

- **SPDIF receiver - Additional features**

There are three exceptions associated with the SPDIF Receivers FIFOs

- full
- under/overflow
- resync

When the "full" condition is set for processor data input registers, the processor should read data from the FIFO, before overflow occurs. When "full" is set, and the FIFO contains e.g. 6 samples, it is acceptable for the software to read first 6 samples from the LEFT address, followed by 6 samples from the RIGHT address, or 6 samples from the RIGHT address, followed by 6 samples from the LEFT address, or 1 sample LEFT, followed by 1 sample RIGHT repeated 6 times. There is no order specified.

The implementation for SPDIF Rx is a double FIFO, one for left and one for right. "full" is set when both FIFOs are full. "underrun, overflow" are set when one of the FIFOs do underrun or do overflow. The resync interrupt means hardware took special action to resynchronize left and right FIFOs.

The FIFO level at which the "full" interrupt is generated, is programmable via the Full Select field in the SPDIFConfigReg register.

#### **Rx FIFO on and Rx FIFO reset.**

Two additional control fields of the SPDIF Rx FIFO are the on/off select and FIFO reset fields.

If on/off select is set to off, all-zero will be read from the FIFO, irrespective of the data received over the SPDIF interface.

If FIFO reset is set, the FIFO is blocked at "1 sample in FIFO". In this, the full interrupt will be on if FullSelect is set to "00". If FullSelect is set to any other value, interrupt will be off. The other interrupts are always off.

### **46.4.1.2 Channel Status Reception**

A total of 48 channel status bits are received in two registers. No interpretation is performed by the SPDIF receiver block.

Channel Status Bits are ordered first bit left. CS-channel MSB bit "0" is located in bit position 23 in the memory-mapped register SPDIFRxCCchannel\_h. CS-channel bit "23" is considered the LSB bit 0 in the register. C-channel bit 24 to 47 is seen as [23:0] bits of register SPDIFRxCCchannel\_l.

#### 46.4.1.2.1 Channel Status Interrupt

When the value of a new SPDIF "CS" channel status frame is loaded in the register, an interrupt is generated. The interrupt is cleared when the processor writes the corresponding bit in the InterruptStat register.

#### 46.4.1.3 User Bit Reception

There are two modes for U Channel reception, CD and non-CD. As is decided by USyncMode (bit 1 of CDText\_Control register).

- **Behavior of U Channel receive interface on incoming CD U Channel Sub-code in SPDIF receiver.**

This mode is selected if USyncMode, bit 1 in register CD Text control is set "1".

The CD sub-code stream embedded into the SPDIF U channel consists of a sequence of packets. Every packet is made up 98 "symbols". The first two symbols of every packet are "sync symbols", the other 96 symbols are "data symbols".

Any sequence found in the SPDIF U channel stream starting with a leading one, followed by 7 information bits, is recognized as a "data symbol". Subsequent data symbols are separated by "pauses". During the "pause", "zero bits" are seen on the SPDIF U channel.

Data symbols are coming in MSB first. The MSB is the leading one.

When a "long pause" is seen between 2 subsequent "data symbols", the SPDIF receiver will assume the reception of one or more "sync symbols". Table below gives details.

**Table 46-3. Sync Control Bits**

Number of U Channel zero bits	Corresponding number of sync symbols
0-1	Unpredictable, not allowed
2-10	0
11-22	1
23-34	2
35-46	3
>45	Unpredictable, not allowed

The recognition of the number of sync symbols derives from the fact that the U channel transmitter in the CD channel decoder will transmit one symbol on average every 12 SPDIF channel bits. On this average rate, there is a maximum tolerance of 5%.

The SPDIF receiver is tolerant of symbol errors. Due to the physical nature of the transmission of the data over the CD disc, not more than 1 out of any 5 consecutive user channel symbols may be in error. The error may cause a change in data value, which is not detected by this interface, or it may cause a data symbol to be seen as a sync symbol, or a sync symbol to be seen as a data symbol. However, not more than 1 out of any 5 consecutive user channel symbols should be affected in this way.

The SPDIF U channel circuitry recognizes the 98-symbol packet structure, and sends the 96 symbol payload to the processor application. The 96 symbol payload is transmitted to the processor via 2 registers:

- The SPDIFRxUChannel register. In this register, data is presented 3 symbols at the time to the processor. Every time 3 new valid symbols, received on the SPDIF U Channel are present, the UChannelRxFull interrupt is asserted. For one 98-symbol packet, 96 symbols are carried across SPDIFRxUChannel. To transfer all this data, 32 UChannelRxFull interrupts are generated.
- The QChannelReceive register. In this register, only the Q bit of the packet is accumulated. Operation is similar to UChannelReceive. Because only Q-bit is transferred, only 96 Q-bits are transferred for any 98-symbol packet. To transfer this data, 4 QChannelRxFull interrupts are generated. When QChannelRxFull occurs, it is coincident with UChannelRxFull. There is only one QChannelRxFull for every 8 UChannelRxFull. The convention is that most significant data is transmitted first, and is left-aligned in the registers.
- Timing regarding packet boundary is extracted by hardware. The last UChannelRxFull corresponding to a given packet should be coincident with the last QChannelRxFull. In this last U, Q channel interrupt, symbols 95-98 are received, Q channel bits 67-98. The interrupts are coincident with UQSyncFound, flagging last symbols of the current frame.
- When the start of the new packet is found before the current packet is complete (less than 98 symbols in the packet), the UQFrameError interrupt is set. The application software should read out UChannelReceive and QchannelReceive registers, discard the value, and assume the start of a new packet.
- As already said, packet sync extraction is tolerant for single-symbol errors. Packet sync detection is based on the recognition of the sequence data-sync-sync-data in the symbol stream, because this is the only syncing sequence that is not affected by single errors. If the sync symbols are not found 98 symbols after the previous

occurrence, it is assumed to be destroyed by channel error, and a new sync symbols is interpolated.

- Normally, only data bytes are passed to the application software. Every databyte will have its most significant bit set. If sync symbols are passed to the application software, they are seen as all-zero symbols. Sync symbols can only end up in the data stream due to channel error.

- **Behavior of U Channel receive interface on incoming non-CD data.**

This mode is selected if UsyncMode, bit 1 in register CD Text control is set '0'.

In non-CD mode, the SPDIF U channel stream is recognized as a sequence of "data symbols". No packet recognition is done.

Any sequence found in the SPDIF U channel stream starting with a leading one, followed by 7 information bits, is recognized as a "data symbol". Subsequent data symbols are separated by "pauses". During the "pause", "zero bits" are seen on the SPDIF U channel.

3 consecutive data symbols seen in the SPDIF U Channel stream are grouped together into the SPDIFRxUChannel register. First symbol is left, last symbol is right aligned. When SPDIFRxUChannel contains 3 new data symbols, UChannelRxFull is asserted.

In this mode, the operation of QchannelRx and associated interrupt QchannelRxFull is reserved, undefined. And the operation of UQFrameError and UQSyncFound is also reserved, undefined.

The U channel is extracted, and output by the SPDIF Rx on SPDIFRxUChannel-Stream.

When incoming SPDIF data parity error or bit error is detected, and if the next SPDIF word for that channel is error-free, the SPDIF word in error is replaced with the average of the previous word and next word. When incoming SPDIF data parity error or bit error is detected, and the next SPDIF word is in error, the previous SPDIF word is repeated.

#### 46.4.1.4 Validity Flag Reception

An interrupt is associated with the Validity flag. (interrupt 16 - SPDIFValNoGood). This interrupt is set every time a frame is seen on the SPDIF interface with the validity bit set to "invalid".

#### 46.4.1.5 SPDIF Receiver Interrupt Exception Definition

Several SPDIF exceptions can trigger an interrupt.

They are:

- Control Status channel change. Set when SPDIFRxChannel\_1 register is updated. The register is updated for every new C-Channel received. The exception is reset on write to InterruptClear register.
- SPDIF Illegal Symbol. Set on reception of illegal symbol during SPDIF receive. Reset by writing register InterruptClear.<sup>1</sup>
- SPDIF bit error. Set on reception of bit error. (Parity bit does not match). Reset on write to InterruptClear register.
- Receive data FIFO full. Set when SPDIF receive data FIFO is full.
- Receive data FIFO underrun/overflow. Set when there is a underrun/overflow on the SPDIF receive data FIFO.
- Receive data FIFO resynchronization. Set when a resynchronization event occurs on the SPDIF receive data FIFO.
- Receive U Channel buffer full. Set when next 24 bits of U channel code are available.
- Receive Q Channel buffer overflow. Set when Q channel buffer overflow.
- Receive U Channel buffer overflow. Set on U channel buffer overflow.
- Receive Q Channel buffer full. Set when next 24 bits of Q channel code are available.
- Receive UQ sync found. Set when UQ channel sync found.
- Receive UQ frame error. Set when UQ frame error found.

### 46.4.1.6 Standards Compliance

The SPDIF interface is compatible with the Tech 3250-E standard of the European Broadcasting Union, except clause 6.3.3 and the IEC60958-3 Ed2 for relevant topics.

Supported input frequency range is 12 KHz up to 96 KHz. (fully compliant) and 96 KHz up to 176 KHz (Can interface with compliant SPDIF transmitter within same cabinet, making reasonable assumptions on jitter added due to interconnecting wire.)

Tolerated jitter on SPDIF input signals are 0.25 bit peak-peak for high frequencies. There is no jitter limit for low frequencies. The user channel extraction in CD mode is capable of coping with single-symbol errors, and still retrieve U channel frames on correct boundaries. This capability is required for reliable reception of CD-Text from some Philips CD channel decoders. This capability was deemed more important than compliance with the IEC60958 annex A.3 standard, and for this reason user channel

---

1. The SPDIF input is a biphasemark modulated signal. The time between any two successive transitions of the SPDIF signal is always 1, 2 or 3 SPDIF symbol periods long. The SPDIF receiver will parse the stream, and split it in so-called symbols. It recognizes s1, s2 and s3 symbols, depending on the length of the symbols. Not all sequences of these symbols are allowed. To give an example, a sequence s2-s1-s1-s1-s2 cannot occur in a no-error SPDIF signal. If the receiver finds such an illegal sequence, the illegal symbol interrupt is set. No corrective action is undertaken. When the interrupt occurs, this means that(a) The SPDIF signal is destroyed by noise (b) The SPDIF frequency changed.

reception is not compliant with IEC60958 annex A.3. However, the interface is capable to receive U channel inserted by a typical CD channel decoder. Also, in this case, it is more robust and tolerant for channel error than what is required by IEC60958 annex A.3.

#### 46.4.1.7 SPDIF PLOCK Detection and Rxclk Output

Using the high speed system clock, the internal DPLL can extract the bit clock (advanced pulse) from the input bitstream. When this internal DPLL is locked, the LOCK bit of PhaseConfig Register will be set, and the SPDIF Lock output pin SPDIF\_LOCK will be asserted.

After DPLL has locked, the pulses are generated, and the average pulse rate is 128 x the sampling frequency. (For a 44.1 KHz input sampling frequency, the average pulse rate = 128 x 44.1 KHz.) The pulse signal is used in the FreqMeas circuit to generate the frequency measurement result.

#### 46.4.1.8 Measuring Frequency of SPDIF\_RxClk

The internal DPLL can extract the bit clock (advanced plus) from the input bitstream. To do that, it is necessary to measure the frequency of the incoming signal in relationship with the system clock (BUS\_CLK).

Associated with it are two registers, PhaseConfig and FreqMeas. The circuit will measure the frequency of the incoming clock as a function of the BUS\_CLK. The circuit is a second-order filter. The output is a value represented by an unsigned number stored in the 24-bit FreqMeas register, giving the frequency of the source as a function of the BUS\_CLK.

$$\text{FreqMeas}[23:0] = \text{FreqMeas\_CLK} / \text{BUS\_CLK} * 2^{10} * \text{GAIN}.$$

For example, if the GAIN is selected as 8\*(2\*\*10) (PhaseConfig[5:3] = 3'b011), the actual result

$$\text{FreqMeas\_CLK} / \text{BUS\_CLK} \text{ is equal to } \text{FreqMeas}[23:0] / 2^{23}.$$

### 46.4.2 SPDIF Transmitter

Audio data for the SPDIF transmitter is provided by processor via the SPDIFTxLeft and SPDIFTxRight registers.

Clocking for SPDIF transmitter is selected through a multiplexer from several clock sources (see [TxClk\\_Source](#) for clock source inputs). The SPDIF transmitter clock source can be divided down as needed using Txclk\_DF. The SPDIF transmitter output can be chosen from either the SPDIF transmitter block, directly from the SPDIF receiver (via the output multiplexer), or disabled.

The SPDIF transmitter generates a SPDIF output bitstream in IEC60958 biphasic mark format, consisting of audio data, channel status.

### 46.4.2.1 Audio Data Transmission

Audio data for the SPDIF transmitter is provided by the processor via SPDIFTxLeft and SPDIFTxRight registers. They send audio data to Tx left and right FIFOs. The Tx left and right FIFOs are also 16-deep and 24-width (equal to the audio data width).

- **SPDIF transmitter data registers - Behavior on overrun, underrun**

The SPDIF Data Transmit registers (SPDIFTxLeft and SPDIFTxRight) have individual FIFOs for left and right channel. As a result, there is always the possibility that left and right FIFOs may go out of sync due to FIFO underruns and FIFO overruns that affect only one part (left or right) of any FIFO. To prevent this from happening, hardware has been added on the device. Two mechanisms to prevent mismatch between the FIFOs are available.

If SPDIF Tx FIFO underruns on the right half of the FIFO, no sample leaves that FIFO (because it was already empty). Special hardware will make sure that the next sample read from the left FIFO will not leave the FIFO (no read strobe is generated). If the underrun occurs on the left half of the FIFO, next read strobe to the right FIFO is blocked.

- **SPDIF transmitter data registers - Automatic resynchronization of FIFOs**

See [Audio Data Reception](#).

- **SPDIFTxLeft, SPDIFTxRight details**

With SPDIF Tx FIFOs three exceptions are associated.

- empty
- under/overrun
- resync

When the empty condition is set for processor data output registers, the processor should write data to the FIFO, before underrun occurs. When empty is set and, for instance, 6 samples need to be written, it is acceptable for the software to write first 6 samples from



the LEFT address, followed by 6 samples from the RIGHT address, or 1 sample LEFT, followed by 1 sample RIGHT repeated 6 times. Left should be written before right. The implementation of all data out FIFOs is a double FIFO, one for left and one for right. Empty is set when both FIFOs are empty. Underrun, overrun are set when one of the FIFOs do underrun or do overrun. Resync is set when the hardware resynchronizes left and right FIFOs.

On receiving underrun, overrun interrupt, synchronization between Left and Right words in the FIFOs may be lost. Synchronization will not be lost when the underrun or overrun comes from the IEC60958 side of the FIFO. If the processor reads or writes more data from, for example, left than from right, synchronization will be lost. If automatic resynchronization is enabled, and if the software obeys the rules to let this work, resynchronization will be automatic.

#### 46.4.2.2 Channel Status Transmission

A total of 48 Consumer channel status bits are transmitted from two registers. Channel Status Bits are ordered first bit left.

CS-channel MSB bit "0" is located in bit position 23 in the memory-mapped register SPDIFTxChannelCons\_h. CS-channel bit "23" is considered bit 0 in the register. C-channel bits 24-47 are seen as MSB-LSB bits of register SPDIFTxChannelCons\_l.

#### 46.4.2.3 Validity Flag Transmission

The validity bit setting is performed via bit 5 of the SPDIF\_SCR register.

### 46.5 SPDIF Memory Map/Register Definition

SPDIF memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
200_4000	SPDIF Configuration Register (SPDIF_SCR)	32	R/W	0000_0400h	<a href="#">46.5.1/2791</a>
200_4004	CDText Control Register (SPDIF_SRCD)	32	R/W	0000_0000h	<a href="#">46.5.2/2793</a>
200_4008	PhaseConfig Register (SPDIF_SRPC)	32	R/W	0000_0000h	<a href="#">46.5.3/2794</a>
200_400C	InterruptEn Register (SPDIF_SIE)	32	R/W	0000_0000h	<a href="#">46.5.4/2795</a>
200_4010	InterruptStat Register (SPDIF_SIS)	32	R	0000_0002h	<a href="#">46.5.5/2797</a>
200_4010	InterruptClear Register (SPDIF_SIC)	32	W	0000_0000h	<a href="#">46.5.6/2799</a>

Table continues on the next page...

## SPDIF memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
200_4014	SPDIFRxLeft Register (SPDIF_SRL)	32	R	0000_0000h	<a href="#">46.5.7/2800</a>
200_4018	SPDIFRxRight Register (SPDIF_SRR)	32	R	0000_0000h	<a href="#">46.5.8/2801</a>
200_401C	SPDIFRxCCChannel_h Register (SPDIF_SRC SH)	32	R	0000_0000h	<a href="#">46.5.9/2801</a>
200_4020	SPDIFRxCCChannel_l Register (SPDIF_SRC SL)	32	R	0000_0000h	<a href="#">46.5.10/2802</a>
200_4024	UchannelRx Register (SPDIF_SRU)	32	R	0000_0000h	<a href="#">46.5.11/2802</a>
200_4028	QchannelRx Register (SPDIF_SRQ)	32	R	0000_0000h	<a href="#">46.5.12/2803</a>
200_402C	SPDIFTxLeft Register (SPDIF_STL)	32	W	0000_0000h	<a href="#">46.5.13/2803</a>
200_4030	SPDIFTxRight Register (SPDIF_STR)	32	W	0000_0000h	<a href="#">46.5.14/2804</a>
200_4034	SPDIFTxCCChannelCons_h Register (SPDIF_STCSCH)	32	R/W	0000_0000h	<a href="#">46.5.15/2804</a>
200_4038	SPDIFTxCCChannelCons_l Register (SPDIF_STCSCL)	32	R/W	0000_0000h	<a href="#">46.5.16/2805</a>
200_4044	FreqMeas Register (SPDIF_SRFM)	32	R	0000_0000h	<a href="#">46.5.17/2805</a>
200_4050	SPDIFTxCk Register (SPDIF_STC)	32	R/W	0002_0F00h	<a href="#">46.5.18/2806</a>

## 46.5.1 SPDIF Configuration Register (SPDIF\_SCR)

Address: 200\_4000h base + 0h offset = 200\_4000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								RxFIFO_Ctrl	RxFIFO_Off_On	RxFIFO_Rst	RxFIFOFull_Sel	RxAutoSync	TxAutoSync	TxFIFOEmpty_Sel	
W																
Reset									0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TxFIFOEmpty_Sel	Reserved	LOW_POWER	soft_reset	TxFIFO_Ctrl	DMA_Rx_En	DMA_TX_En	Reserved	ValCtrl	TxSel	USrc_Sel					
W																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0

**SPDIF\_SCR field descriptions**

Field	Description
31–24 [unimplemented]	This is a 24-bit register the upper byte is unimplemented. This field is reserved.
23 RxFIFO_Ctrl	0 Normal operation 1 Always read zero from Rx data register
22 RxFIFO_Off_On	0 SPDIF Rx FIFO is on 1 SPDIF Rx FIFO is off. Does not accept data from interface
21 RxFIFO_Rst	0 Normal operation 1 Reset register to 1 sample remaining
20–19 RxFIFOFull_Sel	00 Full interrupt if at least 1 sample in Rx left and right FIFOs 01 Full interrupt if at least 4 sample in Rx left and right FIFOs 10 Full interrupt if at least 8 sample in Rx left and right FIFOs 11 Full interrupt if at least 16 sample in Rx left and right FIFO
18 RxAutoSync	0 Rx FIFO auto sync off 1 RxFIFO auto sync on
17 TxAutoSync	0 Tx FIFO auto sync off 1 Tx FIFO auto sync on

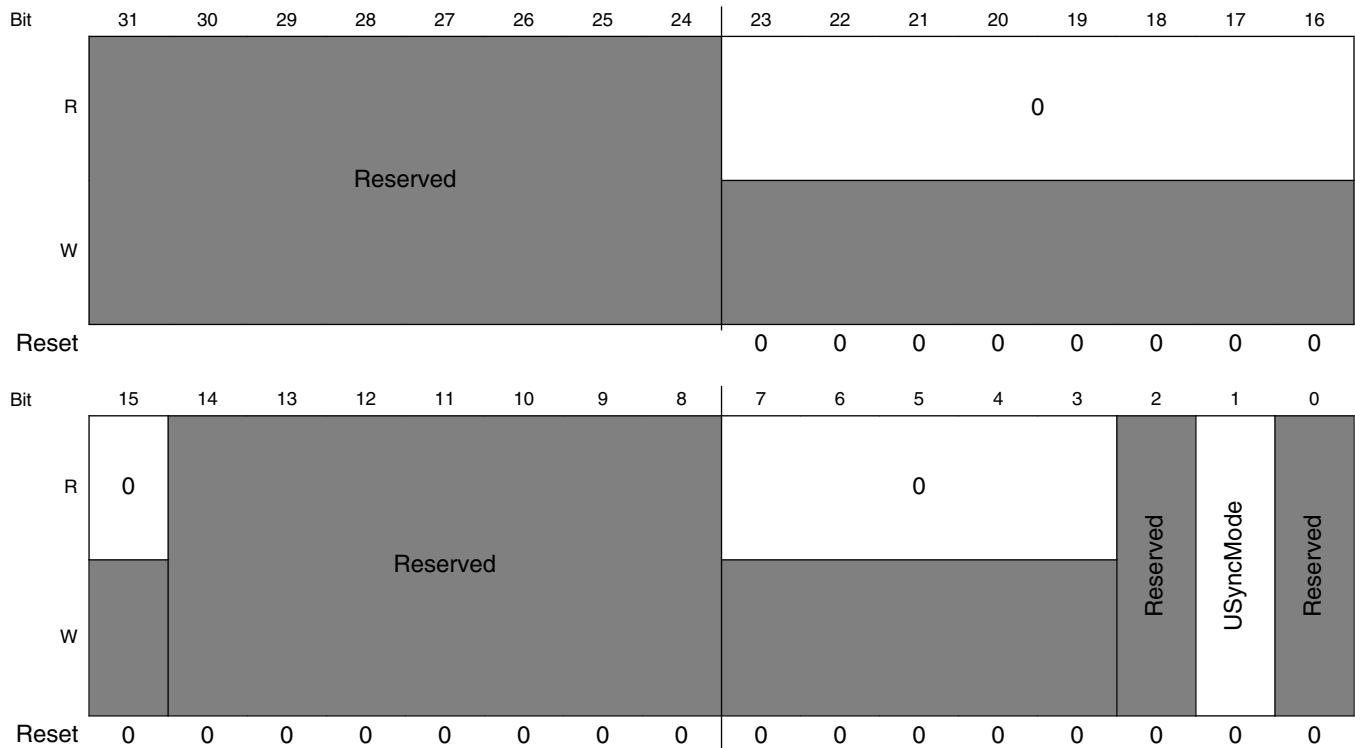
Table continues on the next page...

**SPDIF\_SCR field descriptions (continued)**

Field	Description
16–15 TxFIFOEmpty_Sel	00 Empty interrupt if 0 sample in Tx left and right FIFOs 01 Empty interrupt if at most 4 sample in Tx left and right FIFOs 10 Empty interrupt if at most 8 sample in Tx left and right FIFOs 11 Empty interrupt if at most 12 sample in Tx left and right FIFOs
14 -	This field is reserved. Reserved
13 LOW_POWER	When write 1 to this bit, it will cause SPDIF enter low-power mode. return 1 when SPDIF in Low-Power mode.
12 soft_reset	When write 1 to this bit, it will cause SPDIF software reset. The software reset will last 8 cycles. When in the reset process, return 1 when read. else return 0 when read.
11–10 TxFIFO_Ctrl	00 Send out digital zero on SPDIF Tx 01 Tx Normal operation 10 Reset to 1 sample remaining 11 Reserved
9 DMA_Rx_En	DMA Receive Request Enable (RX FIFO full)
8 DMA_TX_En	DMA Transmit Request Enable (Tx FIFO empty)
7–6 -	This field is reserved. Reserved
5 ValCtrl	0 Outgoing Validity always set 1 Outgoing Validity always clear
4–2 TxSel	000 Off and output 0 001 Feed-through SPDIFIN 101 Tx Normal operation Others Reserved
1–0 USrc_Sel	00 No embedded U channel 01 U channel from SPDIF receive block (CD mode) 10 Reserved 11 U channel from on chip transmitter

## 46.5.2 CDText Control Register (SPDIF\_SRCD)

Address: 200\_4000h base + 4h offset = 200\_4004h

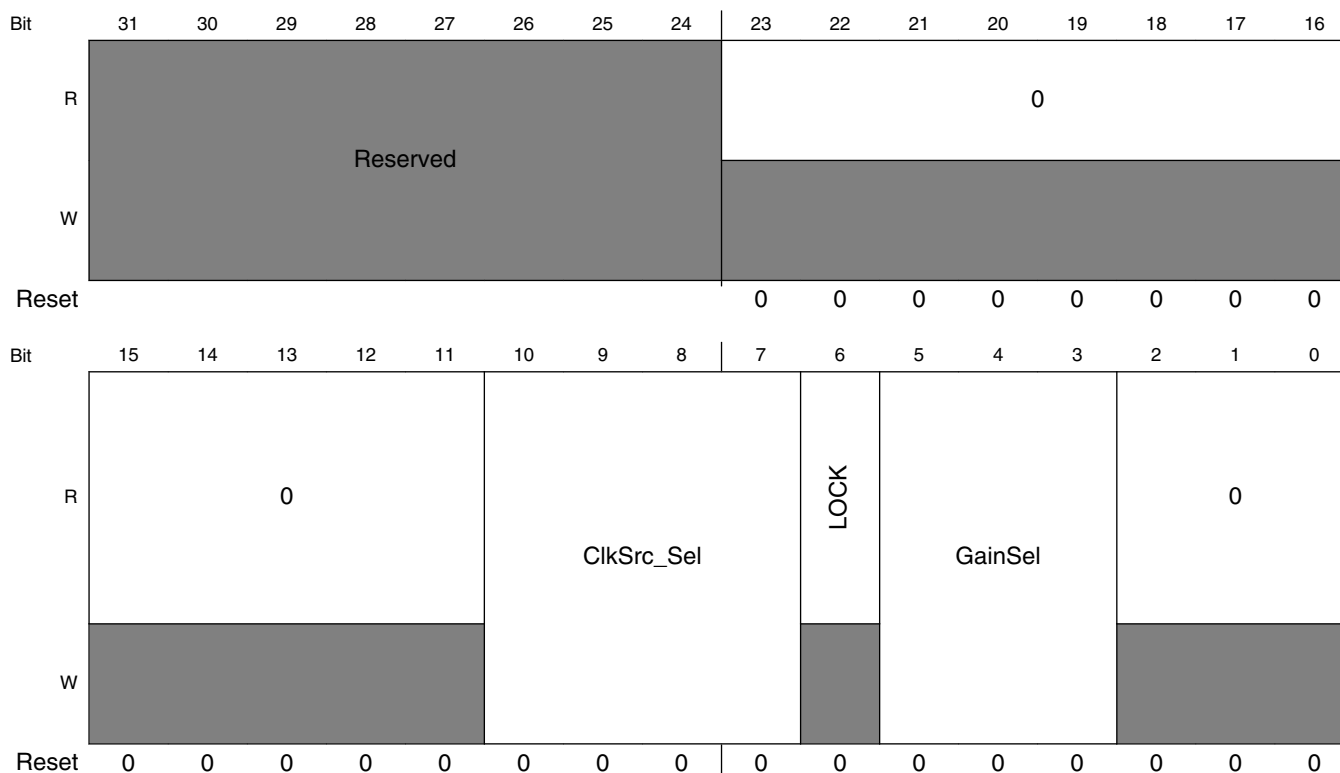


**SPDIF\_SRCD field descriptions**

Field	Description
31–24 [unimplemented]	This is a 24-bit register the upper byte is unimplemented. This field is reserved.
23–15 Reserved	This read-only field is reserved and always has the value 0.
14–8 -	This field is reserved. Reserved
7–3 Reserved	This read-only field is reserved and always has the value 0.
2 -	This field is reserved. Reserved
1 USyncMode	0 Non-CD data 1 CD user channel subcode
0 -	This field is reserved. Reserved

### 46.5.3 PhaseConfig Register (SPDIF\_SRPC)

Address: 200\_4000h base + 8h offset = 200\_4008h



#### SPDIF\_SRPC field descriptions

Field	Description
31–24 [unimplemented]	This is a 24-bit register the upper byte is unimplemented. This field is reserved.
23–11 Reserved	This read-only field is reserved and always has the value 0.
10–7 ClkSrc_Sel	Clock source selection, all other settings not shown are reserved: 0000 if (DPLL Locked) SPDIF_RxCk else REF_CLK_32K (XTALOSC) 0001 if (DPLL Locked) SPDIF_RxCk else tx_clk (SPDIF0_CLK_ROOT) 0011 if (DPLL Locked) SPDIF_RxCk else SPDIF_EXT_CLK 0101 REF_CLK_32K (XTALOSC) 0110 tx_clk (SPDIF0_CLK_ROOT) 1000 SPDIF_EXT_CLK 1100 MLB Clock
6 LOCK	LOCK bit to show that the internal DPLL is locked, read only
5–3 GainSel	Gain selection: 000 24*(2**10)

Table continues on the next page...

**SPDIF\_SRPC field descriptions (continued)**

Field	Description
001	16*(2**10)
010	12*(2**10)
011	8*(2**10)
100	6*(2**10)
101	4*(2**10)
110	3*(2**10)
2–0 Reserved	This read-only field is reserved and always has the value 0.

**46.5.4 InterruptEn Register (SPDIF\_SIE)**

The InterruptEn register (SPDIF\_SIE) provides control over the enabling of interrupts.

Address: 200\_4000h base + Ch offset = 200\_400Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	Reserved								0	Reserved			Lock	TxUnOv	TxResyn	CNew	ValNoGood
W																	
Reset									0	0	0	0	0	0	0	0	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	SymErr	BitErr	Reserved				URxFul	URxOv	QRxFul	QRxOv	UQSync	UQErr	RxFIFOUnOv	RxFIFOResyn	LockLoss	TxEm	RxFIFOFull
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SPDIF\_SIE field descriptions**

Field	Description
31–24 [unimplemented]	This is a 24-bit register the upper byte is unimplemented. This field is reserved.
23 Reserved	This read-only field is reserved and always has the value 0.
22–21 -	This field is reserved. Reserved, for InterruptStat/Clear return zeros when read, for InterruptEn, bit 23 also read zero
20 Lock	SPDIF receiver's DPLL is locked

Table continues on the next page...

**SPDIF\_SIE field descriptions (continued)**

Field	Description
19 TxUnOv	SPDIF Tx FIFO under/overflow
18 TxResyn	SPDIF Tx FIFO resync
17 CNew	SPDIF receive change in value of control channel
16 ValNoGood	SPDIF validity flag no good
15 SymErr	SPDIF receiver found illegal symbol
14 BitErr	SPDIF receiver found parity bit error
13–11 -	This field is reserved. Reserved. Return zeros when read
10 URxFul	U Channel receive register full, can't be cleared with reg. IntClear. To clear it, read from U Rx reg.
9 URxOv	U Channel receive register overrun
8 QRxFul	Q Channel receive register full, can't be cleared with reg. IntClear. To clear it, read from Q Rx reg.
7 QRxOv	Q Channel receive register overrun
6 UQSync	U/Q Channel sync found
5 UQErr	U/Q Channel framing error
4 RxFIFOUnOv	Rx FIFO underrun/overflow
3 RxFIFOResyn	Rx FIFO resync
2 LockLoss	SPDIF receiver loss of lock
1 TxEm	SPDIF Tx FIFO empty, can't be cleared with reg. IntClear. To clear it, write toTx FIFO.
0 RxFIFOFull	SPDIF Rx FIFO full, can't be cleared with reg. IntClear. To clear it, read from Rx FIFO.



### 46.5.5 InterruptStat Register (SPDIF\_SIS)

The InterruptStat (SPDIF\_SIS) register is a read only register that provides the status on interrupt operations.

Address: 200\_4000h base + 10h offset = 200\_4010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								0			Lock	TxUnOv	TxResyn	CNew	ValNoGood
W																
Reset									0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SymErr	BitErr	0			URxFul	URxOv	QRxFul	QRxOv	UQSync	UQErr	RxFIFOUnOv	RxFIFOResyn	LockLoss	TxErn	RxFIFOFull
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

## SPDIF\_SIS field descriptions

Field	Description
31–24 [unimplemented]	This is a 24-bit register the upper byte is unimplemented. This field is reserved.
23–21 Reserved	This read-only field is reserved and always has the value 0.
20 Lock	SPDIF receiver's DPLL is locked
19 TxUnOv	SPDIF Tx FIFO under/overflow
18 TxResyn	SPDIF Tx FIFO resync
17 CNew	SPDIF receive change in value of control channel
16 ValNoGood	SPDIF validity flag no good
15 SymErr	SPDIF receiver found illegal symbol
14 BitErr	SPDIF receiver found parity bit error
13–11 Reserved	This read-only field is reserved and always has the value 0.
10 URxFul	U Channel receive register full, can't be cleared with reg. IntClear. To clear it, read from U Rx reg.
9 URxOv	U Channel receive register overrun
8 QRxFul	Q Channel receive register full, can't be cleared with reg. IntClear. To clear it, read from Q Rx reg.
7 QRxOv	Q Channel receive register overrun
6 UQSync	U/Q Channel sync found
5 UQErr	U/Q Channel framing error
4 RxFIFOUnOv	Rx FIFO underrun/overflow
3 RxFIFOResyn	Rx FIFO resync
2 LockLoss	SPDIF receiver loss of lock
1 TxEm	SPDIF Tx FIFO empty, can't be cleared with reg. IntClear. To clear it, write to Tx FIFO.
0 RxFIFOFull	SPDIF Rx FIFO full, can't be cleared with reg. IntClear. To clear it, read from Rx FIFO.

## 46.5.6 InterruptClear Register (SPDIF\_SIC)

The InterruptClear (SPDIF\_SIC) register is a write only register and is used to clear interrupts.

Address: 200\_4000h base + 10h offset = 200\_4010h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								0							
W																
Reset									0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R			Reserved											Reserved		
W																SymErr
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SPDIF\_SIC field descriptions**

Field	Description
31–24 [unimplemented]	This is a 24-bit register the upper byte is unimplemented. This field is reserved.
23–21 Reserved	This read-only field is reserved and always has the value 0.
20 Lock	SPDIF receiver's DPLL is locked
19 TxUnOv	SPDIF Tx FIFO under/overflow

*Table continues on the next page...*

**SPDIF\_SIC field descriptions (continued)**

Field	Description
18 TxResyn	SPDIF Tx FIFO resync
17 CNew	SPDIF receive change in value of control channel
16 ValNoGood	SPDIF validity flag no good
15 SymErr	SPDIF receiver found illegal symbol
14 BitErr	SPDIF receiver found parity bit error
13–10 -	This field is reserved. Reserved.
9 URxOv	U Channel receive register overrun
8 -	Reserved
7 QRxOv	Q Channel receive register overrun
6 UQSync	U/Q Channel sync found
5 UQErr	U/Q Channel framing error
4 RxFIFOUnOv	Rx FIFO underrun/overrun
3 RxFIFOResyn	Rx FIFO resync
2 LockLoss	SPDIF receiver loss of lock
1–0 -	This field is reserved. Reserved.

**46.5.7 SPDIFRxLeft Register (SPDIF\_SRL)**

SPDIFRxLeft register is an audio data reception register.

Address: 200\_4000h base + 14h offset = 200\_4014h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								RxDataLeft																							
W																																
Reset									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SPDIF\_SRL field descriptions**

Field	Description
31–24 [unimplemented]	This is a 24-bit register the upper byte is unimplemented. This field is reserved.
23–0 RxDataLeft	Processor receive SPDIF data left

**46.5.8 SPDIFRxRight Register (SPDIF\_SRR)**

SPDIFRxRight register is an audio data reception register.

Address: 200\_4000h base + 18h offset = 200\_4018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								RxDataRight																							
W																																
Reset									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SPDIF\_SRR field descriptions**

Field	Description
31–24 [unimplemented]	This is a 24-bit register the upper byte is unimplemented. This field is reserved.
23–0 RxDataRight	Processor receive SPDIF data right

**46.5.9 SPDIFRxCChannel\_h Register (SPDIF\_SRC SH)**

SPDIFRxCChannel\_h register is a channel status reception register.

Address: 200\_4000h base + 1Ch offset = 200\_401Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								RxCChannel_h																							
W																																
Reset									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## SPDIF\_SRC SH field descriptions

Field	Description
31–24 [unimplemented]	This is a 24-bit register the upper byte is unimplemented. This field is reserved.
23–0 RxCChannel_h	SPDIF receive C channel register, contains first 24 bits of C channel without interpretation

## 46.5.10 SPDIFRxCChannel\_I Register (SPDIF\_SRC SL)

SPDIFRxCChannel\_I register is a channel status reception register.

Address: 200\_4000h base + 20h offset = 200\_4020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								RxCChannel_I																							
W																																
Reset	0								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## SPDIF\_SRC SL field descriptions

Field	Description
31–24 [unimplemented]	This is a 24-bit register the upper byte is unimplemented. This field is reserved.
23–0 RxCChannel_I	SPDIF receive C channel register, contains next 24 bits of C channel without interpretation

## 46.5.11 UchannelRx Register (SPDIF\_SR U)

UchannelRx register is a user bits reception register.

Address: 200\_4000h base + 24h offset = 200\_4024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								RxUChannel																							
W																																
Reset	0								0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SPDIF\_SRU field descriptions**

Field	Description
31–24 [unimplemented]	This field is reserved. This is a 24-bit register the upper byte is unimplemented.
23–0 RxUChannel	SPDIF receive U channel register, contains next 3 U channel bytes

**46.5.12 QchannelRx Register (SPDIF\_SRQ)**

QChannelRx register is a user bits reception register.

Address: 200\_4000h base + 28h offset = 200\_4028h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								RxQChannel																							
W																																
Reset									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SPDIF\_SRQ field descriptions**

Field	Description
31–24 [unimplemented]	This is a 24-bit register the upper byte is unimplemented. This field is reserved.
23–0 RxQChannel	SPDIF receive Q channel register, contains next 3 Q channel bytes

**46.5.13 SPDIFTxLeft Register (SPDIF\_STL)**

SPDIFTxLeft register is an audio data transmission register.

Address: 200\_4000h base + 2Ch offset = 200\_402Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																															
W									TxDataLeft																							
Reset									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## SPDIF\_STL field descriptions

Field	Description
31–24 [unimplemented]	This field is reserved. This is a 24-bit register the upper byte is unimplemented.
23–0 TxDataLeft	SPDIF transmit left channel data. It is write-only, and always returns zeros when read

## 46.5.14 SPDIFTxRight Register (SPDIF\_STR)

SPDIFTxRight register is an audio data transmission register.

Address: 200\_4000h base + 30h offset = 200\_4030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																															
W	Reserved								TxDataRight																							
Reset									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

## SPDIF\_STR field descriptions

Field	Description
31–24 [unimplemented]	This field is reserved. This is a 24-bit register the upper byte is unimplemented.
23–0 TxDataRight	SPDIF transmit right channel data. It is write-only, and always returns zeros when read

## 46.5.15 SPDIFTxChannelCons\_h Register (SPDIF\_STCSCH)

SPDIFTxChannelCons\_h register is a channel status transmission register.

Address: 200\_4000h base + 34h offset = 200\_4034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R									TxChannelCons_h																							
W	Reserved																															
Reset									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## SPDIF\_STCSCH field descriptions

Field	Description
31–24 [unimplemented]	This is a 24-bit register the upper byte is unimplemented.

Table continues on the next page...



**SPDIF\_STCSCH field descriptions (continued)**

Field	Description
	This field is reserved.
23–0 TxCChannelCons_h	SPDIF transmit Cons. C channel data, contains first 24 bits without interpretation. When read, it returns the latest data written by the processor

**46.5.16 SPDIFTxCChannelCons\_l Register (SPDIF\_STCSCL)**

SPDIFTxCChannelCons\_l register is a channel status transmission register.

Address: 200\_4000h base + 38h offset = 200\_4038h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R									TxChannelCons_I																							
W	Reserved																															
Reset									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SPDIF\_STCSCL field descriptions**

Field	Description
31–24 [unimplemented]	This is a 24-bit register the upper byte is unimplemented.  This field is reserved.
23–0 TxCChannelCons_l	SPDIF transmit Cons. C channel data, contains next 24 bits without interpretation. When read, it returns the latest data written by the processor

**46.5.17 FreqMeas Register (SPDIF\_SRFM)**

Address: 200\_4000h base + 44h offset = 200\_4044h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								FreqMeas																							
W																																
Reset									0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SPDIF\_SRFM field descriptions**

Field	Description
31–24 [unimplemented]	This is a 24-bit register the upper byte is unimplemented.  This field is reserved.
23–0 FreqMeas	Frequency measurement data

## 46.5.18 SPDIFTxClk Register (SPDIF\_STC)

The SPDIFTxClk Control register includes the means to select the transmit clock and frequency division.

Address: 200\_4000h base + 50h offset = 200\_4050h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								0				SYSCLK_DF			
W																
Reset									0	0	0	0	0	0	1	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SYSCLK_DF						TxClk_Source		tx_all_clk_en	TxClk_DF						
W																
Reset	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0

### SPDIF\_STC field descriptions

Field	Description
31–24 [unimplemented]	This is a 24-bit register the upper byte is unimplemented. This field is reserved.
23–20 Reserved	This read-only field is reserved and always has the value 0.
19–11 SYSCLK_DF	system clock divider factor, 2~512. 0 no clock signal 1 divider factor is 2 ... 511 divider factor is 512
10–8 TxClk_Source	000 REF_CLK_32K input (XTALOSC 32kHz clock) 001 tx_clk input (from SPDIF0_CLK_ROOT. See CCM.) 011 SPDIF_EXT_CLK, from pads 101 ipg_clk input (frequency divided)
7 tx_all_clk_en	Spdif transfer clock enable. When data is going to be transfered, this bit should be set to 1. 0 disable transfer clock. 1 enable transfer clock.
6–0 TxClk_DF	Divider factor (1-128)

Table continues on the next page...

**SPDIF\_STC field descriptions (continued)**

Field	Description
0	divider factor is 1
1	divider factor is 2
...	...
127	divider factor is 128



# Chapter 47

## System Reset Controller (SRC)

### 47.1 SRC Overview

The System Reset Controller (SRC) controls the reset and boot operation of the SoC. It is responsible for the generation of all reset signals and boot decoding.

The reset controller determines the source and the type of reset, such as POR, WARM, COLD, and performs the necessary reset qualification and stretching sequences. Based on the type of reset, the reset logic generates the reset sequence for the entire IC. Whenever the chip is powered on, the reset is issued through SRC\_ONOFF signal and the entire chip is reset.

#### 47.1.1 Features

The SRC includes the following features.

- Receives and handles the resets from all the reset sources
- Resets the appropriate domains based upon the resets sources and the nature of the reset
- Latches the SRC\_BOOT\_MODE pins and common configuration signals from the internal fuse

### 47.2 External Signals

The table found here describes the external signals of SRC.

The following table describes the external signals of SRC:

**Table 47-1. SRC External Signals**

Signal	Description	Pad	Mode	Direction
SRC_BOOT_CFG00	Boot configuration signal	LCD_DAT0	ALT7	I
SRC_BOOT_CFG01	Boot configuration signal	LCD_DAT1	ALT7	I
SRC_BOOT_CFG02	Boot configuration signal	LCD_DAT2	ALT7	I
SRC_BOOT_CFG03	Boot configuration signal	LCD_DAT3	ALT7	I
SRC_BOOT_CFG04	Boot configuration signal	LCD_DAT4	ALT7	I
SRC_BOOT_CFG05	Boot configuration signal	LCD_DAT5	ALT7	I
SRC_BOOT_CFG06	Boot configuration signal	LCD_DAT6	ALT7	I
SRC_BOOT_CFG07	Boot configuration signal	LCD_DAT7	ALT7	I
SRC_BOOT_CFG08	Boot configuration signal	LCD_DAT8	ALT7	I
SRC_BOOT_CFG09	Boot configuration signal	LCD_DAT9	ALT7	I
SRC_BOOT_CFG10	Boot configuration signal	LCD_DAT10	ALT7	I
SRC_BOOT_CFG11	Boot configuration signal	LCD_DAT11	ALT7	I
SRC_BOOT_CFG12	Boot configuration signal	LCD_DAT12	ALT7	I
SRC_BOOT_CFG13	Boot configuration signal	LCD_DAT13	ALT7	I
SRC_BOOT_CFG14	Boot configuration signal	LCD_DAT14	ALT7	I
SRC_BOOT_CFG15	Boot configuration signal	LCD_DAT15	ALT7	I
SRC_BOOT_CFG24	Boot configuration signal	LCD_DAT16	ALT7	I
SRC_BOOT_CFG25	Boot configuration signal	LCD_DAT17	ALT7	I
SRC_BOOT_CFG26	Boot configuration signal	LCD_DAT18	ALT7	I
SRC_BOOT_CFG27	Boot configuration signal	LCD_DAT19	ALT7	I
SRC_BOOT_CFG28	Boot configuration signal	LCD_DAT20	ALT7	I
SRC_BOOT_CFG29	Boot configuration signal	LCD_DAT21	ALT7	I
SRC_BOOT_CFG30	Boot configuration signal	LCD_DAT22	ALT7	I
SRC_BOOT_CFG31	Boot configuration signal	LCD_DAT23	ALT7	I
SRC_BOOT_MODE0	Boot mode signal	BOOT_MODE0	No muxing	I
SRC_BOOT_MODE1	Boot mode signal	BOOT_MODE1	No muxing	I
SRC_ONOFF	ONOFF signal	ONOFF	No muxing	I
SRC_POR_B	Power on reset signal	POR_B	No muxing	I

## 47.3 Clocks

The table found here describes the clock sources for SRC.

Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 47-2. SRC Clocks**

Clock name	Clock Root	Description
ipg_clk	ipg_clk_root	Peripheral clock
ipg_clk_s	ipg_clk_root	Peripheral access clock

## 47.4 Top-level resets, power-up sequence and external supply integration

Information found here defines chip resets, power-up sequence, and external supply integration.

### 47.4.1 Reset and Power-up Flow

The chip presumes the following reset and power-up flow:

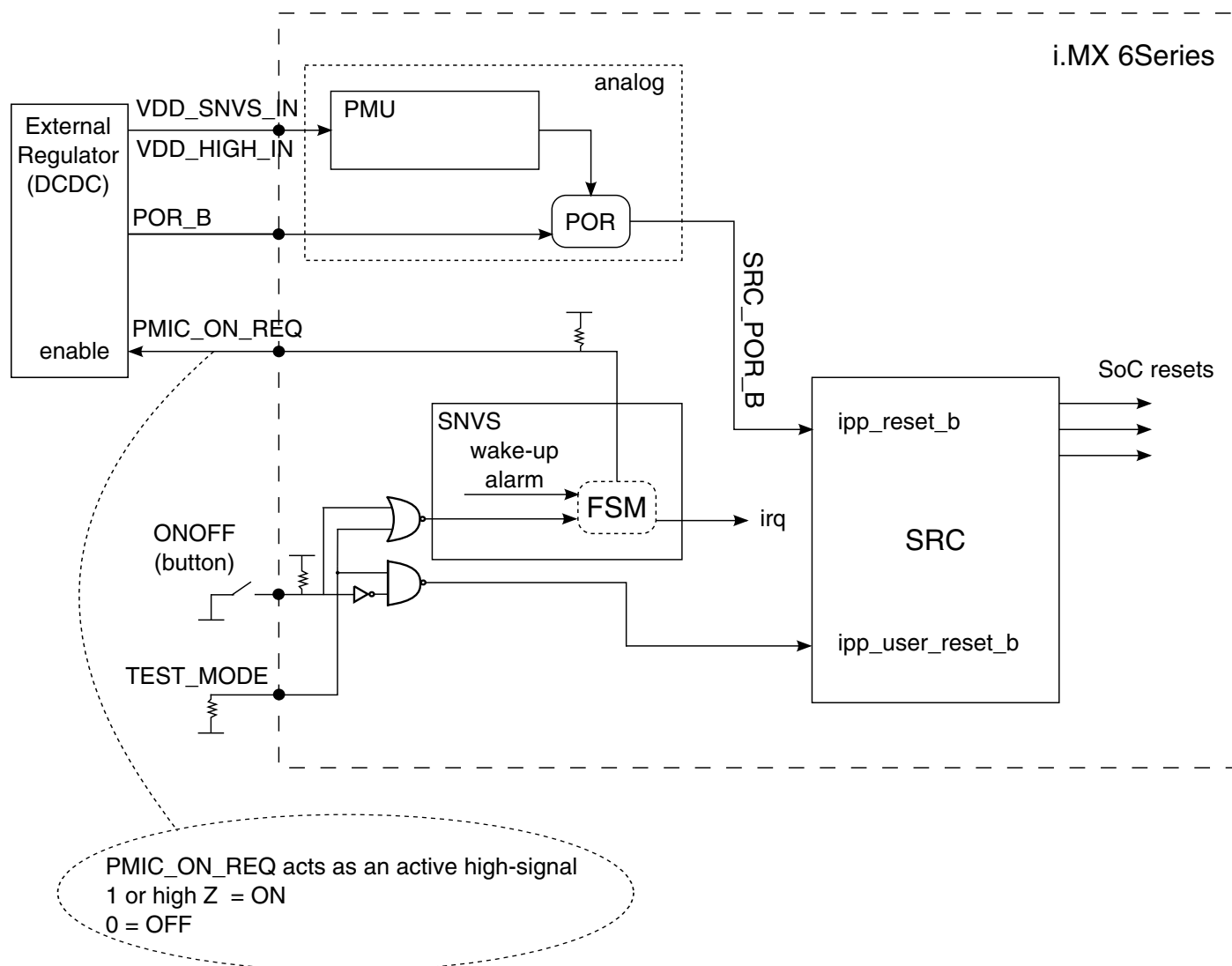
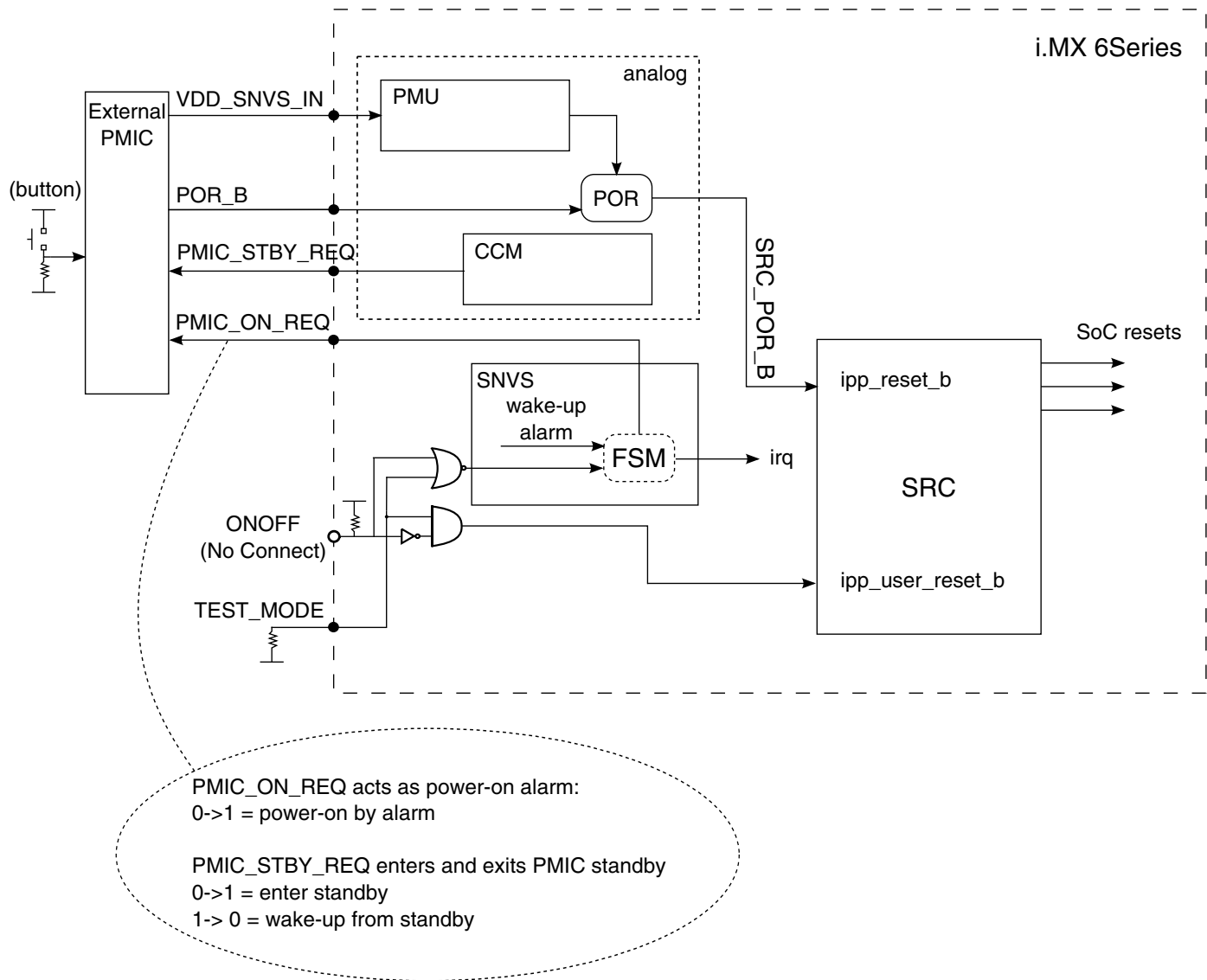


Figure 47-1. Chip reset scheme under PMU control





**Figure 47-2. Chip reset scheme under external PMIC control**

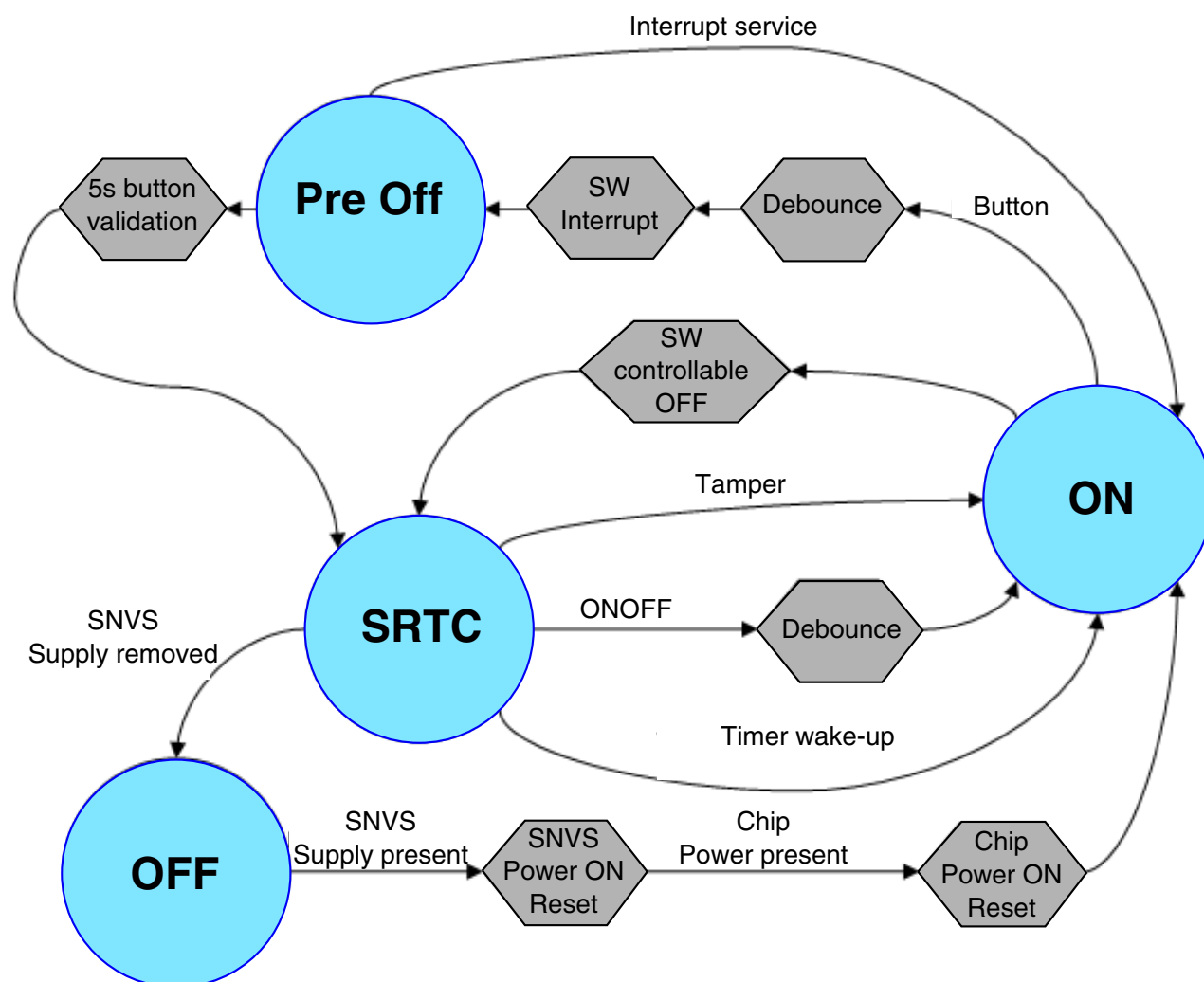


Figure 47-3. Chip on/off state flow diagram

47.4.2 Finite-State Machine (FSM)

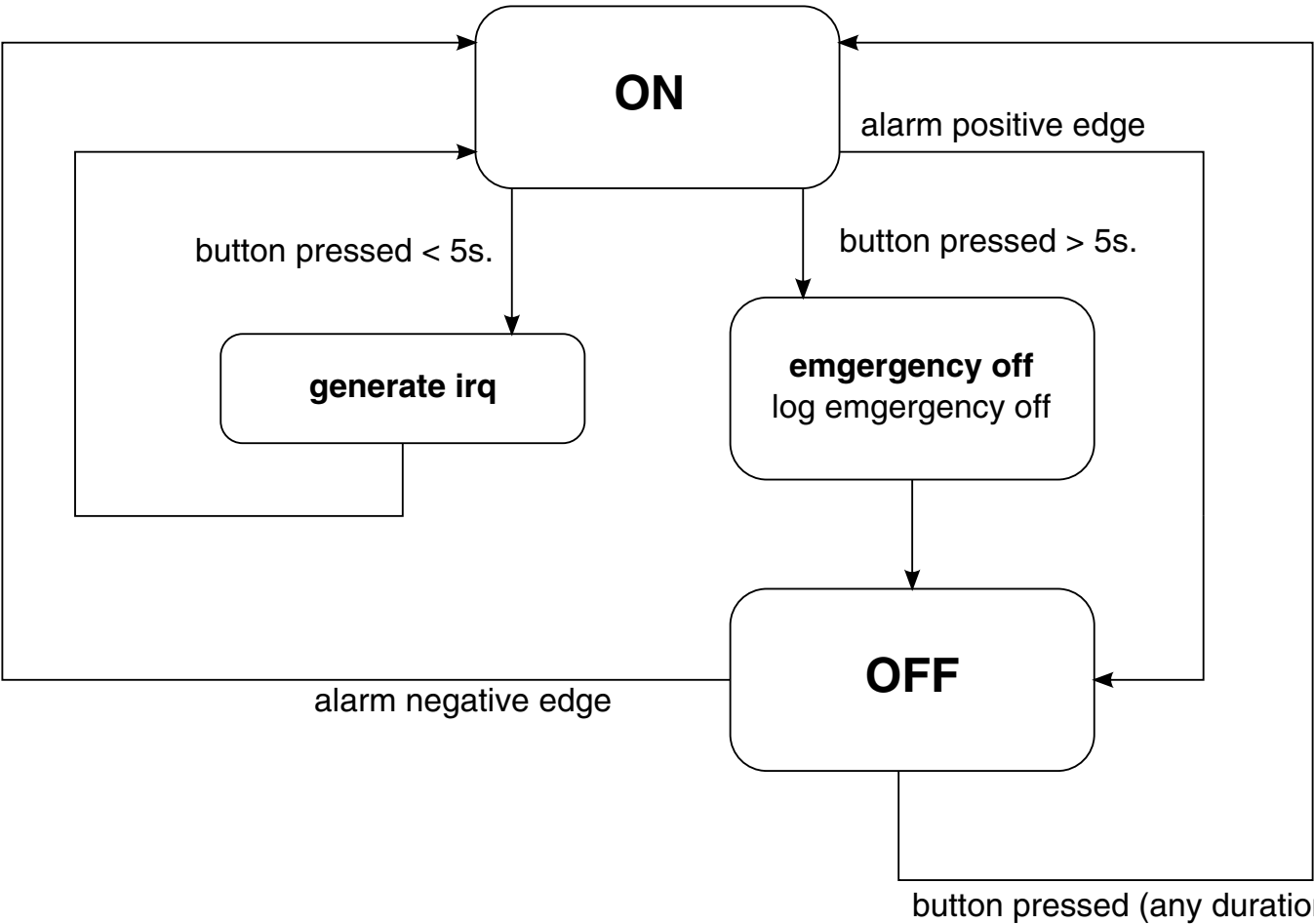


Figure 47-4. FSM

47.4.3 Power mode transitions

Table 47-3. Power mode transitions

Power mode	Configuration with external PMIC	Configuration with internal PMIC
ON, first time	<div>1. Either coin cell or SoC power supply is connected to SNVS.</div> <div>2. When button is pressed, PMIC powers on.</div>	<div>1. Either coin cell or SoC power supply is connected to SNVS.</div> <div>2. When button is pressed, 'state' goes ON, PMIC_ON_REQ goes '1'.</div> <div>3. External regulator is enabled.</div>

Table continues on the next page...

**Table 47-3. Power mode transitions (continued)**

Power mode	Configuration with external PMIC	Configuration with internal PMIC
Normal ON to OFF, by button	<ol style="list-style-type: none"> <li>1. Button is pressed for a short duration on the external PMIC.</li> <li>2. Interrupt request (irq) is sent to SoC from external PMIC.</li> <li>3. SoC is programming PMIC for power off when standby is asserted.</li> <li>4. In CCM STOP mode, Standby is asserted, PMIC gates SoC supplies.</li> </ol>	<ol style="list-style-type: none"> <li>1. SoC button is pressed for a short duration.</li> <li>2. Interrupt request (irq) is sent to SoC from FSM.</li> <li>3. Alarm timer is set up by software routine and started.</li> <li>4. Upon alarm_in assertion to '1', PMIC_ON_REQ goes '0'.</li> <li>5. External regulator goes OFF.</li> </ol>
Emergency ON to OFF, by button	<ol style="list-style-type: none"> <li>1. Button is pressed for an extended time on the external PMIC.</li> <li>2. PMIC is powering off.</li> </ol>	<ol style="list-style-type: none"> <li>1. Button is pressed for longer than 5 seconds on the SoC.</li> <li>2. FSM validates button pressed for 5 seconds.</li> <li>3. Emergency power off is logged, PMIC_ON_REQ goes '0', alarm_mask goes '1'.</li> <li>4. External regulator goes OFF.</li> </ol>
OFF to ON, by button	<ol style="list-style-type: none"> <li>1. Button is pressed on the external PMIC.</li> <li>2. PMIC powers ON.</li> </ol>	<ol style="list-style-type: none"> <li>1. Button is pressed on the SoC.</li> <li>2. PMIC_ON_REQ goes '1', alarm_mask goes '0'.</li> <li>3. External regulator powers ON.</li> </ol>
OFF to ON, by timer alarm	<ol style="list-style-type: none"> <li>1. Timer alarm in SNVS is programmed by software before SoC goes OFF.</li> <li>2. SoC enters OFF mode.</li> <li>3. Upon timer limit, wake up alarm goes '0'. PMIC_ON_REQ goes '1'.</li> <li>4. PMIC receives assertion of PMIC_ON_REQ and wakes up.</li> </ol>	<ol style="list-style-type: none"> <li>1. Timer alarm in SNVS is programmed by software before SoC goes OFF.</li> <li>2. SoC enters OFF mode.</li> <li>3. Upon timer limit, wake up alarm goes '0'. PMIC_ON_REQ goes '1'.</li> <li>4. External regulator is enabled by PMIC_ON_REQ = 1.</li> </ol>

## 47.5 Power-On Reset and power sequencing

This module generates an internal POR\_B signal that is logically AND'ed with any externally applied SRC\_POR\_B signal. The internal POR\_B signal will be held low until all of the following conditions are met:

- 4ms after the external power supply VDDHIGH\_IN is valid
- 1ms after the VDD\_SOC\_CAP supply is valid

The 4ms and 1ms delays are derived from counting the 32kHz RTC clock cycles; the accuracy depends on the accuracy of the RTC. When the RTC crystal is either absent or in the process of powering up, an internal ring oscillator will be the source of RTC, which is not as accurate as the crystal.

### 47.5.1 External POR using SRC\_POR\_B

If the external SRC\_POR\_B signal is used to control the processor POR, SRC\_POR\_B must remain low (asserted) until the VDD\_ARM\_CAP and VDD\_SOC\_CAP supplies are stable.

### 47.5.2 Internal POR

If the external SRC\_POR\_B signal is not used (always held high or left unconnected), the processor defaults to the internal POR function (PMU controls generation of the POR based on the power supplies).

If the internal POR function is used, the following power supply requirements must be met:

- VDD\_ARM\_IN and VDD\_SOC\_IN may be supplied from the same source, or
- VDD\_SOC\_IN can be supplied before VDD\_ARM\_IN with a maximum delay of 1 ms.

## 47.6 Functional Description

### 47.6.1 Reset Control

This section details the reset control of this device.

#### NOTE

SNVS resets are discussed in the *Multimedia Applications Processor Security Reference Manual*.

#### 47.6.1.1 Reset inputs and outputs

The reset control logic receives reset requests from all potential reset sources. All the immediate sources of reset are directly passed to the reset stretching block, whereas the resets requiring qualification are passed on to the reset qualification logic before they are sent to the reset stretching block.

All reset inputs and outputs are described in the following figure:

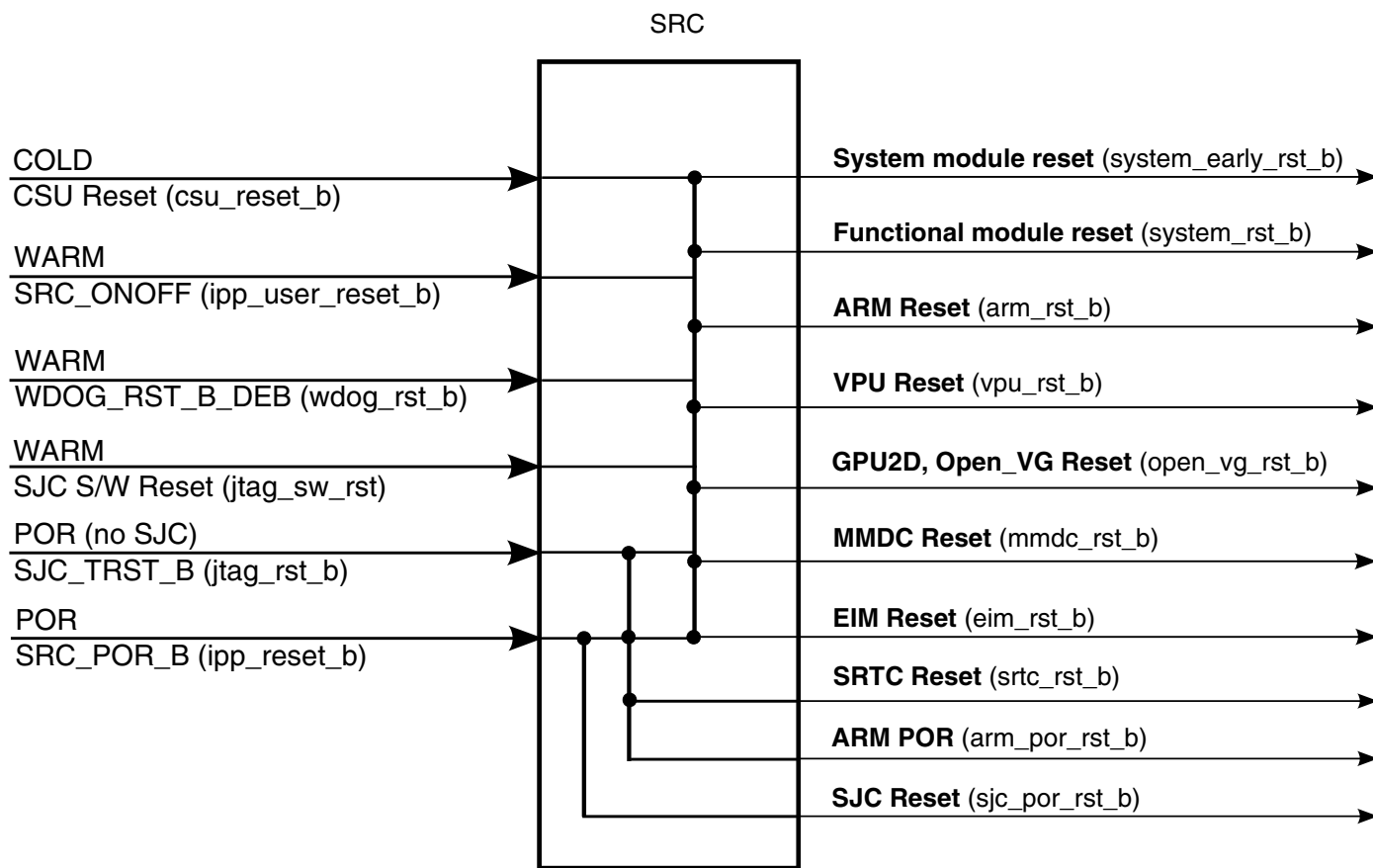


Figure 47-5. SRC inputs and outputs

The reset types and modules they affect are shown in [Table 47-4](#). As there is no chip POR, the POR\_B is used to reset the entire chip including test logic and JTAG modules.

### NOTE

All resets are expected to be active low except jtag\_sw\_rst.

Table 47-4. SRC reset functionality

SoC Modules	POR	COLD	WARM
System modules (PLLs, fuses, etc)	yes	yes	yes
Functional modules	yes	yes	yes
ARM	yes	yes	yes
ARM SoC	yes	yes	yes
VPU	yes	yes	yes
GPU2D	yes	yes	yes
MMDC	yes	yes	yes
ARM POR	yes	no	no
ARM debug	yes	no	no
SJC	yes	no	no

Table continues on the next page...

**Table 47-4. SRC reset functionality (continued)**

SoC Modules	POR	COLD	WARM
SRTC	yes	no	no

The reset priorities are POR (strongest), COLD and WARM (weakest). If a stronger reset is asserted during the sequence of a weaker reset, then the weaker sequence will be overridden, and the stronger reset sequence will commence. There is no priority within a reset type (POR, etc). If a reset is asserted during the reset sequence of the same type, the reset sequence will be interrupted and restarted.

The following lists the functionality of each of these reset outputs:

- system\_early\_rst\_b - Resets the system modules that need to start first as CCM, OCOTP\_CTRL, FUSEBOX, MMDC, etc.
- system\_rst\_b - Resets functional modules
- arm\_rst\_b - Resets ARM module (on regular system reset)
- vpu\_rst\_b - Resets VPU module (on regular system reset)
- gpu\_rst\_b - Resets GPU3D module (on regular system reset)
- open\_vg\_rst\_b - Resets Open\_VG and GPU2D modules (on regular system reset)
- mmhc\_rst\_b - Resets MMDC
- arm\_por\_rst\_b - Resets ARM por input
- arm\_soc\_rst\_b - Reset for ARM SOC
- arm\_dbg\_rst\_b - Reset debug logic of ARM
- test\_logic\_rst\_b - Reset test logic (IOMUXC, DAP)
- sjc\_por\_rst\_b - Reset to SJC
- srtc\_rst\_b - Resets SRTC

### NOTE

It is assumed that each reset source will deassert after its assertion, either due to reset generated to the system from SRC, or by negation of the reset source (if it came from an external source to the chip). In the latter case, the reset source is assumed to be held for at least 2 XTALI clocks so it can be sampled by SRC.

## 47.6.1.2 Reset Handling

### 47.6.1.2.1 Reset Qualification

The reset qualification logic qualifies the reset source before sending it out to the chip as a valid reset. WARM resets are in this category. All remaining reset sources are immediate resets and are acknowledged by the reset circuitry the moment they are asserted.

WARM resets are not immediate resets. WARM resets do reset the CCM, the source of MMDC clock. So, if a WARM reset were to immediately reset the CCM, then the MMDC clock would be shut off and this may cause the MMDC data to be lost. During normal mode of operation of the chip, the protocol that is followed before shutting off the MMDC clock is that an `mmdc_dvfs_req` signal is sent to MMDC and only after the MMDC sends an acknowledge signal, `mmdc_dvfs_ack`, is the clock to the MMDC gated off.

However, the implication here is that a valid WARM reset source condition will not be able to cause a chip reset until the MMDC sends the acknowledge signal (`mmdc_dvfs_ack`). For example, a JTAG reset event has occurred but the JTAG reset will not be serviced until the `mmdc_dvfs_ack` signal is received. So, essentially all WARM reset sources depend on the MMDC providing the `mmdc_dvfs_ack` acknowledge signal before the reset is performed. When the MMDC is not used, `mmdc_dvfs_ack` is defaulted high.

The occurrence of WARM reset results in the assertion of `warm_reset` signal before the system resets are asserted to indicate to the MMDC that the reset occurred is a WARM reset.

A reset source is updated in the Reset Status Register (`SRC_SRSR`) when it becomes valid, provided it is asserted for the minimum amount of time after asserting. So, all immediate resets are immediately updated in the Status Register (`SRC_SRSR`). WARM resets would be updated when the `mmdc_dvfs_ack` signal is received from the MMDC.

Once the reset is qualified, depending on the source of the reset, internal resets are asserted appropriately.

### 47.6.1.2.2 Reset Sequence and De-Assertion

The `SRC_ONOFF` will assert immediately after any reset source is recognized (except for the case of WARM reset when MMDC needs to answer `mmdc_dvfs_ack` first). After all of the reset sources are released, the SRC will start the following set of events depending on the type of reset that occurred.



### 47.6.1.2.3 POR (SRC\_POR\_B)

SRC\_POR\_B is an external reset signal. When the chip is powered up, the reset signal is passed through the POR\_B pin indicating power-up sequence. The SRC resets the entire chip including the JTAG (SJC) module. All SRC registers will be reset during the POR sequence.

As soon as SRC\_POR\_B occurs, all resets are asserted and the entire chip is reset by SRC. The SRC\_POR\_B is stretched for 2 XTALI cycles and the stretching sequence takes place after 2 XTALI clocks of POR\_B pin deassertion.

The sjc\_por\_rst\_b and srtc\_rst\_b signals are deasserted together with SRC\_POR\_B signal. Those outputs are also deasserted after the stretching of SRC\_POR\_B has deasserted.

Once the above resets deassert, system\_early\_rst\_b reset is deasserted after 2 XTALI clocks. The system\_early\_rst\_b is used for the CCM and PLL-IPs to start generating PLL clock outputs and the system root clocks.

When the system root clocks are ready, the CCM will assert system\_clk\_ready signal. This signal is generated during the start sequence in the CCM and it involves the preparation of the PLLs to generate clock roots for functional operation.

SRC then enables OCOTP\_CTRL and fusebox clocks, so that fuses can be loaded to OCOTP\_CTRL.

- SRC will prepare the boot information and then de assert mmdc\_rst\_b.
- SRC will wait 8 ipg clock cycles, and then SRC will enable MMDC clocks.
- SRC will wait 8 XTALI clocks to allow MMDC to generate fixed external clock to external memory SDRAM.
- SRC will enable VPU and GPU clocks.
- After 8 ipg cycles, SRC will disable VPU, Open\_VG and GPU clocks.
- After 8 ipg cycles, resets to all modules will be de-asserted (system\_rst\_b, vpu\_rst\_b, gpu\_rst\_b, open\_vg\_rst\_b).
- After 8 ipg cycles, system clocks will be enabled (en\_system\_clk).

### 47.6.1.2.4 COLD RESET

The sequence is similar to SRC\_POR\_B except the memory repair operation is not performed.

Once the reset source deasserts, system\_early\_rst\_b reset is deasserted after at least 2 XTALI clocks. The system\_early\_rst\_b is used for the CCM and PLL-IPs to start generating PLL clock outputs and the system root clocks.

Once the system root clocks are ready, the CCM will assert `system_clk_ready` signal. This signal is generated during the start sequence in the CCM and it involves the preparation of the PLLs to generate clock roots for functional operation. See CCM for more information.

Once `system_clk_ready` arrives at the SRC, it will enable `OCOTP_CTRL` and fusebox clocks, so that fuses can be loaded to `OCOTP_CTRL`. `OCOTP_CTRL` will notify with `iim_ready_flag` once the fusebox loading finishes.

- SRC will prepare the boot information and then deassert `mmdc_rst_b`.
- SRC will wait 8 ipg clock cycles, and then SRC will enable MMDC clocks.
- SRC will wait 8 XTALI clocks to allow MMDC to generate fixed external clock to external memory SDRAM.
- SRC will enable VPU and GPU clocks so that reset will penetrate this module.
- After 8 ipg cycles, SRC will dissable VPU, Open\_VG and GPU clocks.
- After 8 ipg cycles resets to all modules will be deasserted (`system_rst_b`, `vpu_rst_b`, `gpu_rst_b`, `open_vg_rst_b`).
- After 8 ipg cycles, system clocks will be enabled (`en_system_clk`).

#### **47.6.1.2.5 WARM RESET**

WARM reset will be enabled only if `SRC_SCR[warm_reset_enable]` bit is programmed. Otherwise, all WARM reset sources will generate a COLD reset. This bit will be reset only by a POR.

A WARM reset is similar to a COLD reset except that before the reset is sent, a signal to MMDC is asserted `mmdc_dvfs_req` (generates DVFS assertion to MMDC) to request to prepare MMDC to a WARM reset, finishing the transactions placing MMDC in self-refresh. Another signal will be asserted to MMDC (`warm_reset`) that will wrap the WARM reset sequence and will notify MMDC that a WARM reset is in process.

One of the sources of the WARM reset is `SRC_ONOFF`. If this is the case, it is qualified for 4 XTALI edges. The WARM reset is initiated immediately after 4 XTALI edges. The system does not come out of reset until the the `SRC_ONOFF` is released.

In case the handshake mechanism with MMDC is stuck, meaning that no `mmdc_dvfs_ack` is recieved, COLD reset will be generated after a number of XTALI clocks. The number of XTALI clocks is defined in register the `SRC_SCR[warm_rst_bypass_count]` bitfield (default of this bitfield is 16 XTALI counts.)

The following is a basic description of the WARM reset sequence:

1. ARM sets `SRC_SCR[warm_reset_enable]` bit to enable the WARM Reset functionality. If this bit is not set, all WARM reset sources will result in COLD reset.
2. Assertion of one of the WARM reset sources.

3. The reset source is qualified in the SRC.
4. If `mmdc_dvfs_ack` signal is low, then SRC triggers the MMDC to switch to self-refresh mode using `mmdc_dvfs_req` signal. This is done through the CCM to combine with the DVFS sent from the CCM in case of frequency change of MMDC.
5. Wait for `mmdc_dvfs_ack` signal from the MMDC. If no ack is received during `warm_rst_bypass_count` number of XTALI clocks, COLD reset will be generated.
6. Assert `warm_reset` signal to MMDC.
7. SRC asserts system resets

The deassertion sequence is exactly the same as in the Cold Reset except waiting for 8 XTALIs for MMDC to generate fixed external clock to external memory MMDC. This stage is not needed in WARM reset since MMDC is held in self-refresh in WARM reset and there is no need to reconfigure it when exiting WARM reset.

## WARM BOOT

Software can save any needed information in the memory before initiating a WARM reset. In this case, software will set `SRC_SRSR[warm_boot]` bit before initiating WARM reset. After the system returns to run mode, the `warm_boot` bit will still be set, indicating the software that data was saved in memory and can be reused.

### NOTE

`mmdc_dvfs_req` and acknowledge during WARM reset can be masked in the CCM by configuration of register `CCDR[17:16]`.

## 47.6.2 Parallel Reset Requests

SRC will follow the following rules in the case of parallel reset requests:

1. The order of strength of resets is POR - strongest, cold - medium, warm - weakest.
2. If a stronger reset is asserted during weaker reset sequence, then the stronger reset will take over and the stronger reset process will commence. The following cases fall into this category:
  - POR reset request in the middle of cold or warm reset process - the cold or warm reset process will be stopped and the POR sequence will start.
  - COLD reset request in the middle of warm reset process - the warm reset process will be stopped and the cold sequence will start.
3. If a weaker reset is asserted during stronger reset sequence, then the stronger reset sequence will continue without interference. If at the end of the stronger reset process the weaker request is still asserted then the weaker sequence will commence. The following cases fall into this category:

- COLD or WARM reset requests in the middle of POR reset process - the POR process will continue without interference.
  - WARM reset request in the middle of COLD reset process - the COLD process will continue without interference.
4. If a similar reset request is asserted during the process of reset handling, then the process of reset handling will start over (with the same process). The following cases fall into this category:
- POR reset request in the middle of POR reset process - the POR process will start over.
  - COLD reset request in the middle of COLD reset process - the COLD process will start over.

There is one exception to this category: WARM reset request in the middle of WARM reset process. In this case, the new WARM reset process cannot restart because MMDC is reset and there can't be a handshake with MMDC if it is reset. In this case the first WARM reset will continue, and only if the second WARM reset is still asserted after the first one has finished, the WARM sequence will start again.

## **47.6.3 Boot Mode Control**

### **47.6.3.1 BOOT\_MODE Pin Latching**

The exact boot sequence is controlled by the values of the BOOT\_MODE pins on this device.

The value of the BOOT\_MODE pins will be latched after the OCOTP\_CTRL asserts the fuse read completion flag. After latching, the values of the BOOT\_MODE pins are used to determine the booting options of the core as described in the SRC\_SBMRx registers.

The boot mode general purpose bits can be provided to the SRC from either e-fuses or GPIO signals. The gpio\_bt\_sel e-fuse defines the source to be used to derive the boot information. When gpio\_bt\_sel is set, e-fuses are used. When cleared, GPIO signals are used.

The boot information is provided in SRC\_SBMR1 register. The figure below shows the selection of boot mode information.

#### **NOTE**

BOOT\_MODE[1:0] inputs of SRC are connected to  
BOOT\_MODE[1:0] pins.

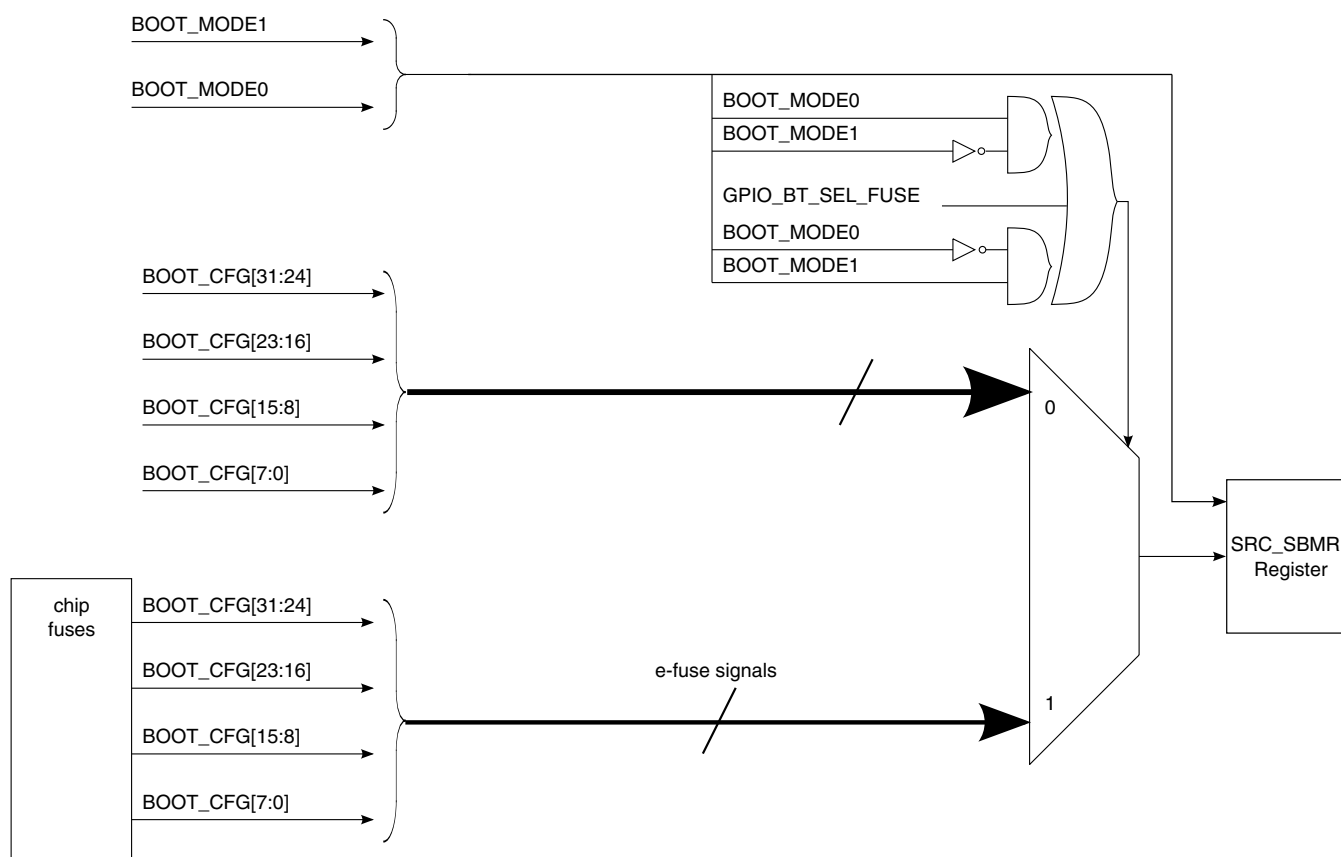


Figure 47-6. Boot mode information

## 47.7 SRC Memory Map/Register Definition

### SRC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20D_8000	SRC Control Register (SRC_SCR)	32	R/W	0000_0521h	<a href="#">47.7.1/2827</a>
20D_8004	SRC Boot Mode Register 1 (SRC_SBMR1)	32	R	0000_0000h	<a href="#">47.7.2/2830</a>
20D_8008	SRC Reset Status Register (SRC_SRSR)	32	R/W	0000_0001h	<a href="#">47.7.3/2830</a>
20D_8014	SRC Interrupt Status Register (SRC_SISR)	32	R	0000_0000h	<a href="#">47.7.4/2833</a>
20D_8018	SRC Interrupt Mask Register (SRC_SIMR)	32	R/W	0000_001Fh	<a href="#">47.7.5/2835</a>
20D_801C	SRC Boot Mode Register 2 (SRC_SBMR2)	32	R	0000_0000h	<a href="#">47.7.6/2837</a>
20D_8020	SRC General Purpose Register 1 (SRC_GPR1)	32	R/W	0000_0000h	<a href="#">47.7.7/2838</a>
20D_8024	SRC General Purpose Register 2 (SRC_GPR2)	32	R/W	0000_0000h	<a href="#">47.7.8/2838</a>
20D_8028	SRC General Purpose Register 3 (SRC_GPR3)	32	R/W	0000_0000h	<a href="#">47.7.9/2839</a>

Table continues on the next page...

## SRC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20D_802C	SRC General Purpose Register 4 (SRC_GPR4)	32	R/W	0000_0000h	<a href="#">47.7.10/2839</a>
20D_8030	SRC General Purpose Register 5 (SRC_GPR5)	32	R/W	0000_0000h	<a href="#">47.7.11/2839</a>
20D_8034	SRC General Purpose Register 6 (SRC_GPR6)	32	R/W	0000_0000h	<a href="#">47.7.12/2840</a>
20D_8038	SRC General Purpose Register 7 (SRC_GPR7)	32	R/W	0000_0000h	<a href="#">47.7.13/2840</a>
20D_803C	SRC General Purpose Register 8 (SRC_GPR8)	32	R/W	0000_0000h	<a href="#">47.7.14/2840</a>
20D_8040	SRC General Purpose Register 9 (SRC_GPR9)	32	R/W	0000_0000h	<a href="#">47.7.15/2841</a>
20D_8044	SRC General Purpose Register 10 (SRC_GPR10)	32	R/W	0000_0000h	<a href="#">47.7.16/2841</a>

## 47.7.1 SRC Control Register (SRC\_SCR)

The Reset control register (SCR), contains bits that control operation of the reset controller.

Address: 20D\_8000h base + 0h offset = 20D\_8000h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	0						dbg_rst_msk_pg	0				cores_dbg_rst	0			core0_dbg_rst	0
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	0		core0_rst		Reserved	eim_rst	mask_wdog_rst				warm_rst_bypass_count		sw_open_vg_rst	Reserved	sw_vpu_rst	sw_gpu_rst	warm_reset_enable
W																	
Reset	0	0	0	0	0	1	0	1	0	0	1	0	0	0	0	1	

**SRC\_SCR field descriptions**

Field	Description
31–26 Reserved	This read-only field is reserved and always has the value 0.
25 dbg_rst_msk_pg	Do not assert debug resets after power gating event of core 0 do not mask core debug resets (debug resets will be asserted after power gating event) 1 mask core debug resets (debug resets won't be asserted after power gating event)
24–22 Reserved	This read-only field is reserved and always has the value 0.
21 cores_dbg_rst	Software reset for debug of arm platform only. <b>NOTE:</b> This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared.

*Table continues on the next page...*

## SRC\_SCR field descriptions (continued)

Field	Description
	0 do not assert arm platform debug reset 1 assert arm platform debug reset
20–18 Reserved	This read-only field is reserved and always has the value 0.
17 core0_dbg_rst	Software reset for core0 debug only. <b>NOTE:</b> This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert core0 debug reset 1 assert core0 debug reset
16–14 Reserved	This read-only field is reserved and always has the value 0.
13 core0_rst	Software reset for core0 only. <b>NOTE:</b> This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. 0 do not assert core0 reset 1 assert core0 reset
12 -	This field is reserved. Reserved
11 eim_rst	EIM reset is needed in order to reconfigure the eim chip select. The software reset bit must de-asserted. The eim chip select configuration should be updated. The software bit must be re-asserted since this is not self-refresh.
10–7 mask_wdog_rst	Mask wdog_rst_b source. If these 4 bits are coded from A to 5 then, the wdog_rst_b input to SRC will be masked and the wdog_rst_b will not create a reset to the chip. <b>NOTE:</b> During the time the WDOG event is masked using SRC logic, it is likely that the WDOG Reset Status Register (WRSR) bit 1 (which indicates a WDOG timeout event) will get asserted. software / OS developer must prepare for this case. Re-enabling the WDOG is possible, by unmasking it in SRC, though it must be preceded by servicing the WDOG. However, for the case that the event has been asserted, the status bit (WRSR bit-1) will remain asserted, regardless of servicing the WDOG module.  (Hardware reset is the only way to cause the de-assertion of that bit). any other code will be coded to 1010 i.e. wdog_rst_b is not masked  0101 wdog_rst_b is masked 1010 wdog_rst_b is not masked (default)
6–5 warm_rst_bypass_count	Defines the XTALI cycles to count before bypassing the MMDC acknowledge for WARM reset. If the MMDC acknowledge will not be asserted before this counter has elapsed, then a COLD reset will be initiated.  00 Counter not to be used - system will wait until MMDC acknowledge until it is asserted. 01 Wait 16 XTALI cycles before changing WARM reset to a COLD reset. 10 Wait 32 XTALI cycles before changing WARM reset to a COLD reset. 11 Wait 64 XTALI cycles before changing WARM reset to a COLD reset

Table continues on the next page...



## SRC\_SCR field descriptions (continued)

Field	Description
4 sw_open_vg_rst	<p>Software reset for open_vg</p> <p><b>NOTE:</b> This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. Software can determine that the reset has finished once this bit is cleared. Software can also configure SRC to generate interrupt once the software has finished. Please refer to SRC_SISR register for details.</p> <p><b>NOTE:</b> The reset process will involve 8 open_vg cycles before negating the open_vg reset, to allow reset assertion to propagate into open_vg.</p> <p>0 do not assert open_vg reset 1 assert open_vg reset</p>
3 -	<p>This field is reserved.</p> <p>Reserved</p>
2 sw_vpu_rst	<p>Software reset for VPU</p> <p><b>NOTE:</b> This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. Software can determine that the reset has finished once this bit is cleared. Software can also configure SRC to generate interrupt once the software has finished. Please refer to SRC_SISR register for details.</p> <p><b>NOTE:</b> The reset process will involve 8 VPU cycles before negating the VPU reset, to allow reset assertion to propagate into VPU.</p> <p>0 do not assert VPU reset 1 assert VPU reset</p>
1 sw_gpu_rst	<p>Software reset for GPU</p> <p><b>NOTE:</b> This is a self clearing bit. Once it is set to 1, the reset process will begin, and once it finishes, this bit will be self cleared. Software can determine that the reset has finished once this bit is cleared. Software can also configure SRC to generate interrupt once the software has finished. Please refer to SRC_SISR register for details.</p> <p><b>NOTE:</b> The reset process will involve 8 GPU cycles before negating the GPU reset, to allow reset assertion to propagate into GPU.</p> <p>0 do not assert GPU reset 1 assert GPU reset</p>
0 warm_reset_enable	<p>WARM reset enable bit. WARM reset will be enabled only if warm_reset_enable bit is set. Otherwise all WARM reset sources will generate COLD reset.</p> <p>0 WARM reset disabled 1 WARM reset enabled</p>

## 47.7.2 SRC Boot Mode Register 1 (SRC\_SBMR1)

The Boot Mode register (SBMR) contains bits that reflect the status of Boot Mode Pins of the chip. The reset value is configuration dependent (depending on boot/fuses/IO pads).

Address: 20D\_8000h base + 4h offset = 20D\_8004h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	BOOT_CFG4[7:0]								BOOT_CFG3[7:0]								BOOT_CFG2[7:0]								BOOT_CFG1[7:0]							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### SRC\_SBMR1 field descriptions

Field	Description
31–24 BOOT_CFG4[7:0]	Refer to fusemap.
23–16 BOOT_CFG3[7:0]	Refer to fusemap.
15–8 BOOT_CFG2[7:0]	Refer to fusemap.
7–0 BOOT_CFG1[7:0]	Refer to fusemap.

## 47.7.3 SRC Reset Status Register (SRC\_SRSR)

The SRSR is a write to one clear register which records the source of the reset events for the chip. The SRC reset status register will capture all the reset sources that have occurred. This register is reset on ipp\_reset\_b. This is a read-write register.

For bit[6-0] - writing zero does not have any effect. Writing one will clear the corresponding bit. The individual bits can be cleared by writing one to that bit. When the system comes out of reset, this register will have bits set corresponding to all the reset sources that occurred during system reset. Software has to take care to clear this register by writing one after every reset that occurs so that the register will contain the information of recently occurred reset.

Address: 20D\_8000h base + 8h offset = 20D\_8008h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															warm_boot
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								jtag_sw_rst	jtag_rst_b	wdog_rst_b	ipp_user_reset_b	csu_reset_b	0	ipp_reset_b	
W									w1c	w1c	w1c	w1c	w1c		w1c	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

## SRC\_SRSR field descriptions

Field	Description
31–17 Reserved	This read-only field is reserved and always has the value 0.
16 warm_boot	<p>WARM boot indication shows that WARM boot was initiated by software. This indicates to the software that it saved the needed information in the memory before initiating the WARM reset. In this case, software will set this bit to '1', before initiating the WARM reset. The warm_boot bit should be used as indication only after a warm_reset sequence. Software should clear this bit after warm_reset to indicate that the next warm_reset is not performed with warm_boot. Please refer to <a href="#">Reset Sequence and De-Assertion</a> for details on warm_reset.</p> <p>0 WARM boot process not initiated by software. 1 WARM boot initiated by software.</p>
15–7 Reserved	This read-only field is reserved and always has the value 0.
6 jtag_sw_rst	JTAG software reset. Indicates whether the reset was the result of software reset from JTAG.

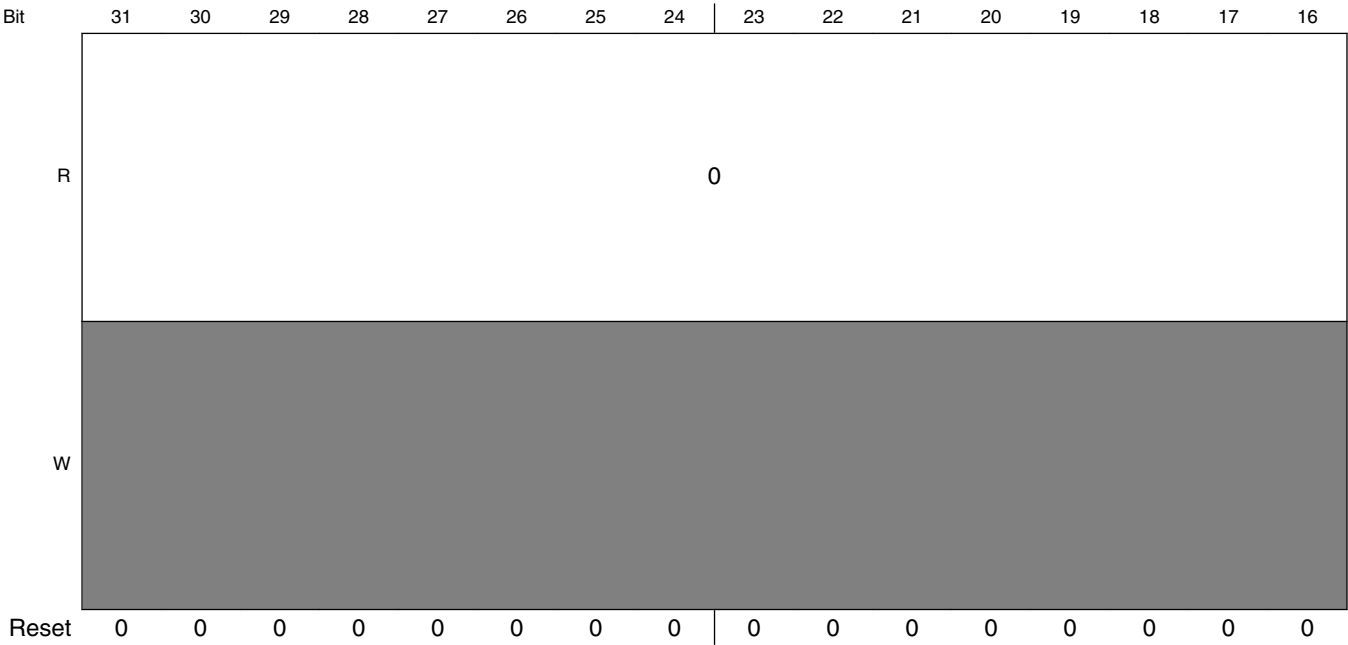
Table continues on the next page...

## SRC\_SRSR field descriptions (continued)

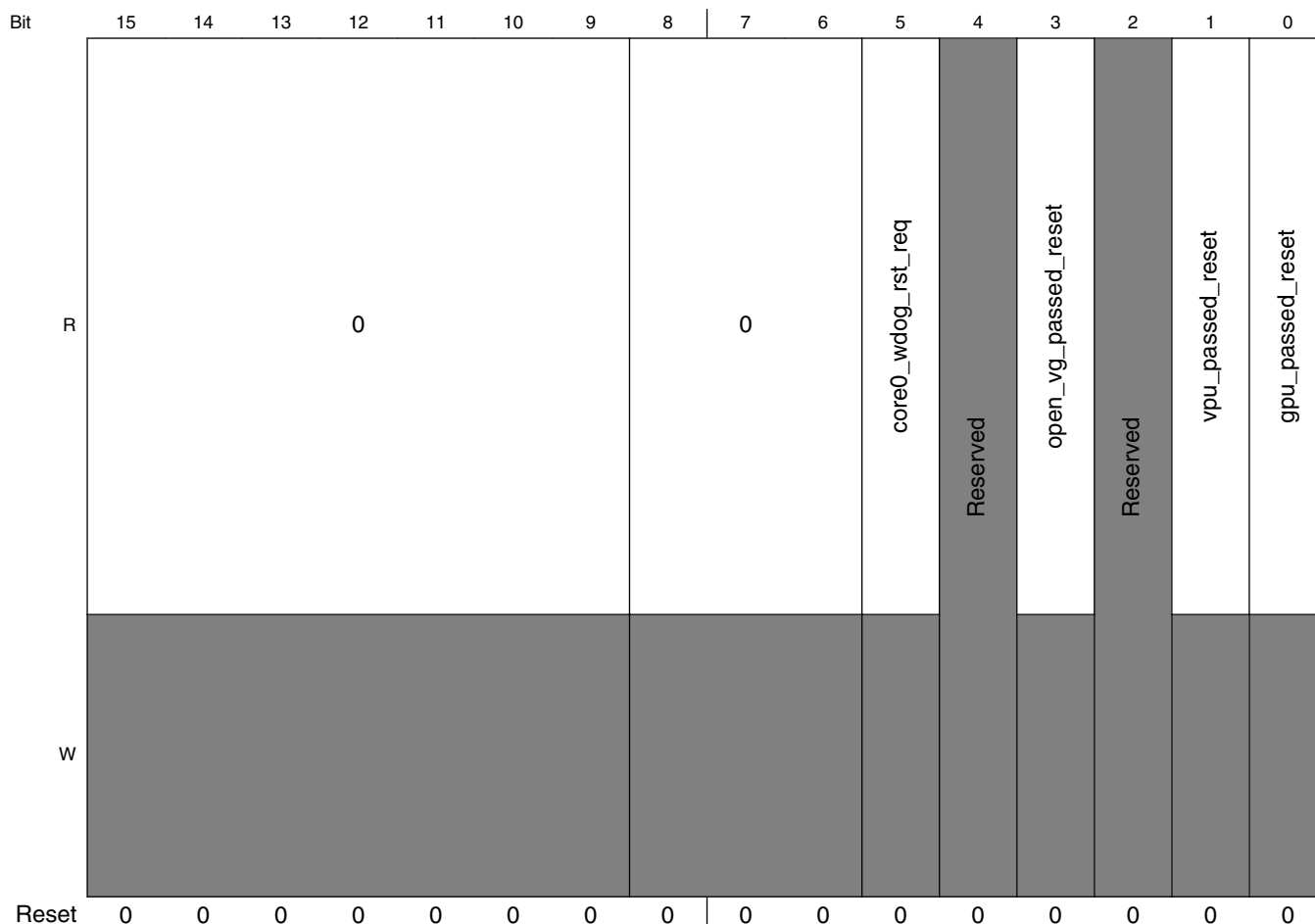
Field	Description
	0 Reset is not a result of software reset from JTAG. 1 Reset is a result of software reset from JTAG.
5 jtag_rst_b	HIGH - Z JTAG reset. Indicates whether the reset was the result of HIGH-Z reset from JTAG. 0 Reset is not a result of HIGH-Z reset from JTAG. 1 Reset is a result of HIGH-Z reset from JTAG.
4 wdog_rst_b	IC Watchdog Time-out reset. Indicates whether the reset was the result of the watchdog time-out event. 0 Reset is not a result of the watchdog time-out event. 1 Reset is a result of the watchdog time-out event.
3 ipp_user_reset_b	Indicates whether the reset was the result of the ipp_user_reset_b qualified reset. 0 Reset is not a result of the ipp_user_reset_b qualified as COLD reset event. 1 Reset is a result of the ipp_user_reset_b qualified as COLD reset event.
2 csu_reset_b	Indicates whether the reset was the result of the csu_reset_b input. <b>NOTE:</b> If case the csu_reset_b occurred during a WARM reset process, during the phase that ipg_clk is not available yet, then the occurrence of CSU reset will not be reflected in this bit. 0 Reset is not a result of the csu_reset_b event. 1 Reset is a result of the csu_reset_b event.
1 Reserved	This read-only field is reserved and always has the value 0.
0 ipp_reset_b	Indicates whether reset was the result of ipp_reset_b pin (Power-up sequence) 0 Reset is not a result of ipp_reset_b pin. 1 Reset is a result of ipp_reset_b pin.

### 47.7.4 SRC Interrupt Status Register (SRC\_SISR)

Address: 20D\_8000h base + 14h offset = 20D\_8014h



## SRC Memory Map/Register Definition



**SRC\_SISR field descriptions**

Field	Description
31–9 Reserved	This read-only field is reserved and always has the value 0.
8–6 Reserved	This read-only field is reserved and always has the value 0.
5 core0_wdog_rst_req	WDOG reset request from core0. Read-only status bit.
4 -	This field is reserved. Reserved
3 open_vg_passed_reset	Interrupt generated to indicate that open_vg passed software reset and is ready to be used 0 interrupt generated not due to open_vg passed reset 1 interrupt generated due to open_vg passed reset
2 -	This field is reserved. Reserved
1 vpu_passed_reset	Interrupt generated to indicate that VPU passed software reset and is ready to be used 0 interrupt generated not due to VPU passed reset 1 interrupt generated due to VPU passed reset

Table continues on the next page...

**SRC\_SISR field descriptions (continued)**

Field	Description
0 gpu_passed_reset	Interrupt generated to indicate that GPU passed software reset and is ready to be used
0	interrupt generated not due to GPU passed reset
1	interrupt generated due to GPU passed reset

**47.7.5 SRC Interrupt Mask Register (SRC\_SIMR)**

Address: 20D\_8000h base + 18h offset = 20D\_8018h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								Reserved				mask_open_vg_passed_reset	Reserved	mask_vpu_passed_reset	mask_gpu_passed_reset
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	1

**SRC\_SIMR field descriptions**

Field	Description
31–5 Reserved	This read-only field is reserved and always has the value 0.
4 -	This field is reserved. Reserved
3 mask_open_vg_passed_reset	mask interrupt generation due to open_vg passed reset 0 do not mask interrupt due to open_vg passed reset - interrupt will be created 1 mask interrupt due to open_vg passed reset
2 -	This field is reserved. Reserved

*Table continues on the next page...*

**SRC\_SIMR field descriptions (continued)**

Field	Description
1 mask_vpu_ passed_reset	mask interrupt generation due to VPU passed reset 0 do not mask interrupt due to VPU passed reset - interrupt will be created 1 mask interrupt due to VPU passed reset
0 mask_gpu_ passed_reset	mask interrupt generation due to GPU passed reset 0 do not mask interrupt due to GPU passed reset - interrupt will be created 1 mask interrupt due to GPU passed reset



### 47.7.6 SRC Boot Mode Register 2 (SRC\_SBMR2)

The Boot Mode register (SBMR), contains bits that reflect the status of Boot Mode Pins of the chip. The default values for those bits depends on the values of pins/fuses during reset sequence, hence the question mark on their default value.

Address: 20D\_8000h base + 1Ch offset = 20D\_801Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	0						BMOD[1:0]		0								
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	0										BT_FUSE_SEL			DIR_BT_DIS	0	SEC_CONFIG[1:0]	
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

## SRC\_SBMR2 field descriptions

Field	Description
31–26 Reserved	This read-only field is reserved and always has the value 0.
25–24 BMOD[1:0]	Refer to fusemap.
23–5 Reserved	This read-only field is reserved and always has the value 0.
4 BT_FUSE_SEL	BT_FUSE_SEL (connected to gpio bt_fuse_sel)
3 DIR_BT_DIS	Refer to fusemap.
2 Reserved	This read-only field is reserved and always has the value 0.
1–0 SEC_CONFIG[1:0]	Refer to fusemap.

## 47.7.7 SRC General Purpose Register 1 (SRC\_GPR1)

Address: 20D\_8000h base + 20h offset = 20D\_8020h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	<div>PERSISTENT_ENTRY0</div>																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

## SRC\_GPR1 field descriptions

Field	Description
31–0 PERSISTENT_ENTRY0	Holds entry function for core0 for waking-up from low power mode. The SRC ensures that the register value will persist across system resets.

## 47.7.8 SRC General Purpose Register 2 (SRC\_GPR2)

Address: 20D\_8000h base + 24h offset = 20D\_8024h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SRC\_GPR2 field descriptions**

Field	Description
31–0 PERSISTENT_ARG0	Holds argument of entry function for core0 for waking-up from low power mode. The SRC ensures that the register value will persist across system resets.

**47.7.9 SRC General Purpose Register 3 (SRC\_GPR3)**

Address: 20D\_8000h base + 28h offset = 20D\_8028h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																	-															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SRC\_GPR3 field descriptions**

Field	Description
31–0 -	Read/write general purpose bits used to store an arbitrary value.

**47.7.10 SRC General Purpose Register 4 (SRC\_GPR4)**

Address: 20D\_8000h base + 2Ch offset = 20D\_802Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																	-															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SRC\_GPR4 field descriptions**

Field	Description
31–0 -	Read/write general purpose bits used to store an arbitrary value.

**47.7.11 SRC General Purpose Register 5 (SRC\_GPR5)**

Address: 20D\_8000h base + 30h offset = 20D\_8030h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																	-															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

## SRC\_GPR5 field descriptions

Field	Description
31–0 -	Read/write general purpose bits used to store an arbitrary value.

## 47.7.12 SRC General Purpose Register 6 (SRC\_GPR6)

Address: 20D\_8000h base + 34h offset = 20D\_8034h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## SRC\_GPR6 field descriptions

Field	Description
31–0 -	Read/write general purpose bits used to store an arbitrary value.

## 47.7.13 SRC General Purpose Register 7 (SRC\_GPR7)

Address: 20D\_8000h base + 38h offset = 20D\_8038h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## SRC\_GPR7 field descriptions

Field	Description
31–0 -	Read/write general purpose bits used to store an arbitrary value.

## 47.7.14 SRC General Purpose Register 8 (SRC\_GPR8)

Address: 20D\_8000h base + 3Ch offset = 20D\_803Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SRC\_GPR8 field descriptions**

Field	Description
31–0 -	Read/write general purpose bits used to store an arbitrary value.

**47.7.15 SRC General Purpose Register 9 (SRC\_GPR9)**

Reserved for Internal Use.

Address: 20D\_8000h base + 40h offset = 20D\_8040h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SRC\_GPR9 field descriptions**

Field	Description
31–0 -	This field is reserved. Reserved.

**47.7.16 SRC General Purpose Register 10 (SRC\_GPR10)**

Address: 20D\_8000h base + 44h offset = 20D\_8044h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	-	PERSIST_	-													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	-															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## SRC\_GPR10 field descriptions

Field	Description
31 -	Read/write bit, for general purpose <b>NOTE:</b> Reset only by POR
30 PERSIST_ SECONDARY_ BOOT	This bit identifies which image must be used - primary and secondary. Used only for boot modes that support redundant boot.
29–0 -	Read/write bits, for general purpose <b>NOTE:</b> Reset only by POR

# Chapter 48

## Synchronous Serial Interface (SSI)

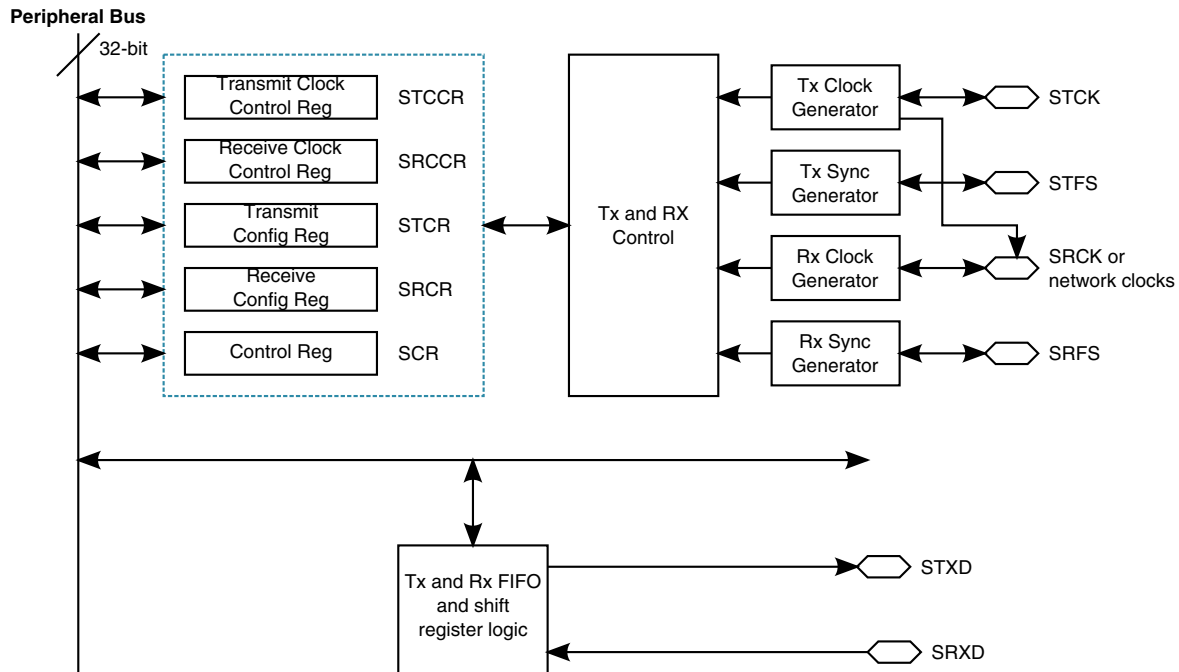
### 48.1 Overview

This block guide presents the Synchronous Serial Interface (SSI), and discusses the architecture, the programming model, the operating modes, and initialization of SSI.

The SSI is a full-duplex, serial port that allows the chip to communicate with a variety of serial devices. These serial devices can be standard CODer-DECoder (CODECs), Digital Signal Processors (DSPs), microprocessors, peripherals, and popular industry audio codecs that implement the inter-IC sound bus standard (I2S) and Intel AC97 standard.

SSI is typically used to transfer samples in a periodic manner. The SSI consists of independent transmitter and receiver sections with independent clock generation and frame synchronization.

The [Figure 48-1](#) illustrates the organization of the SSI. It consists of control registers to set up the port, status register, separate transmit and receive circuits with FIFO registers, and separate serial clock and frame sync generation for the transmit and receive sections. The second set of Tx and Rx FIFOs, replicates the logic used for the first set of FIFOs.



**Figure 48-1. SSI Block Diagram**

### 48.1.1 Features

The SSI includes the following features:

- Independent (asynchronous) or shared (synchronous) transmit and receive sections with separate or shared internal/external clocks and frame syncs, operating in Master or Slave mode.
- Normal mode operation using frame sync
- Network mode operation allowing multiple devices to share the port with as many as thirty-two time slots
- Gated Clock mode operation requiring no frame sync
- 2 sets of Transmit and Receive FIFOs. Each of the four FIFOs is 15x32 bits. The two sets of Tx/Rx FIFOs can be used in Network mode to provide 2 independent channels for transmission and reception (this mode is named as two-channel mode in the following descriptions)
- Programmable data interface modes such as I2S, LSB, MSB aligned
- Programmable word length (8, 10, 12, 16, 18, 20, 22 or 24 bits)
- Program options for frame sync and clock generation
- Programmable I2S modes (Master, Slave or Normal). Maximum audio sampling rate is 196kHz. (Note that maximum sampling rate depends on IPG frequency.) Minimum audio sampling rate is 8kHz. Network clock (as an oversampling clock to external device) available as output from SRCK in I2S Master mode



- AC97 support. Max frame rate is 48kHz. Min frame rate is 8kHz.
- Completely separate clock and frame sync selections for the receive and transmit sections. In AC97 standard, the clock is taken from an external source and frame sync is generated internally.
- SSI's system clock (generated inside CCM) can be used in I2S Transmitter Master mode. This system clock is also available as source clock for output SRCK in master mode, when operated in sync mode.
- Programmable internal clock divider
- Time Slot Mask Registers for reduced ARM platform overhead (for both Tx and Rx)
- SSI power-down feature

### 48.1.2 Modes of Operation

SSI has the following basic operating modes.

- **Normal Mode** : Asynchronous protocol, Synchronous protocol
  - **Normal Mode Transmit**
  - **Normal Mode Receive**
- **Network Mode** : Asynchronous protocol, Synchronous protocol
  - **Network Mode Transmit**
  - **Network Mode Receive**
- **Gated Clock Mode** : Synchronous protocol only
- **I2S Mode**
- **AC97 Mode**
  - **AC97 Fixed Mode** (SSI.SACNT[1]=0)
  - **AC97 Variable Mode** (SSI.SACNT[1]=1)

## 48.2 External Signal Description

## 48.2.1 Signals Overview

The Synchronous Serial Interface (SSI) can be connected directly to the external pins or through the Digital Audio Multiplexer (AUDMUX). Refer to the AUDMUX chapter for programming details of the various multiplexing options.

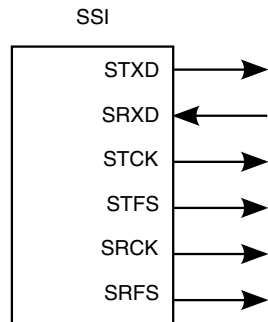
**Table 48-1. Off-Chip Block Signals**

Name	I/O	Function	Reset State	Pull up
SRCK	I/O	Serial Receive Clock. SRCK can be used as either an input or an output. This clock signal is used by the receiver in asynchronous mode and is always continuous. During synchronous mode, the STCK port is used instead for clocking in data. In SSI synchronous modes, this port can be used as an output for the network clock (oversampling clock). In I2S master mode, this signal can be used to output the network clock to an external CODEC.	0	Passive
SRFS	I/O	Serial Receive Frame Sync. The SRFS port can be used as either an input or an output. The frame sync is used by the receiver to synchronize the transfer of data. The frame sync signal can be one bit or one word in length and can occur one bit before the transfer of data or right at the transfer of data. If SRFS is configured as an input, the external device should drive SRFS during the rising edge of STCK or SRCK.	0	Passive
SRXD	I	Serial Receive Data. The SRXD port is an input and is used to bring serial data into the Receive Data Shift Register.	-	-
STCK	I/O	Serial Transmit Clock. The STCK port can be used as either an input or an output. This clock signal is used by the transmitter and can be either continuous or gated. During Gated Clock mode, data on the STCK port is valid only during the transmission of data, otherwise it is pulled to the inactive state. In Synchronous mode, this port is used by both the transmit and receive sections.	0	Passive
STFS	I/O	Serial Transmit Frame Sync. The STFS port can be used as either an input or an output. The frame sync is used by the transmitter to synchronize the transfer of data. The frame sync signal can be one bit or one word in length and can occur one bit before the transfer of data or right at the transfer of data. In Synchronous mode, this port is used by both the transmit and receive sections. In Gated Clock mode, frame sync signals are not used. If STFS is configured as an input, the external device should drive STFS during the rising edge of STCK if TSCKP is +ve edge triggered. The external device should drive STFS during the falling edge of STCK if TSCKP is -ve edge triggered.	0	Passive
STXD	O	Serial Transmit Data. The STXD port is an output and transmits data from the Serial Transmit Shift Register. The STXD port is an output port when data is being transmitted and is disabled between data word transmissions and on the trailing edge of the bit clock after the last bit of a word is transmitted.	0	Passive

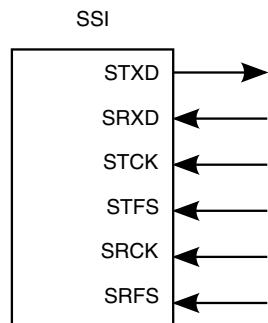
The following figure shows the main SSI configurations. These ports support all transmit and receive functions with continuous or gated clock as shown.

### NOTE

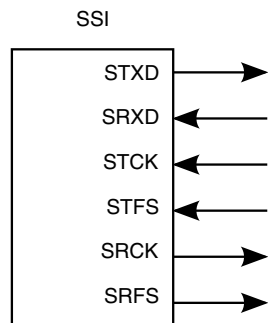
Gated clock implementations do not require the use of the frame sync ports (STFS and SRFS).



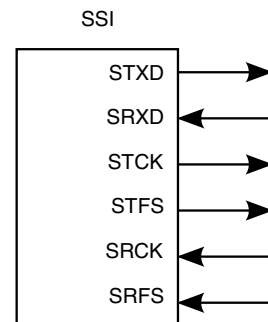
SSI Internal Continuous Clock for TX/RX (RXDIR = 1, TXDIR = 1, RFDIR = 1, TFDIR = 1, SYN = 0)



SSI External Continuous Clock for TX/RX (RXDIR = 0, TXDIR = 0, RFDIR = 0, TFDIR = 0, SYN = 0)



SSI Internal Continuous Clock for RX (RXDIR = 1, TXDIR = 0, RFDIR = 1, TFDIR = 0, SYN = 0)  
SSI External Continuous Clock for TX

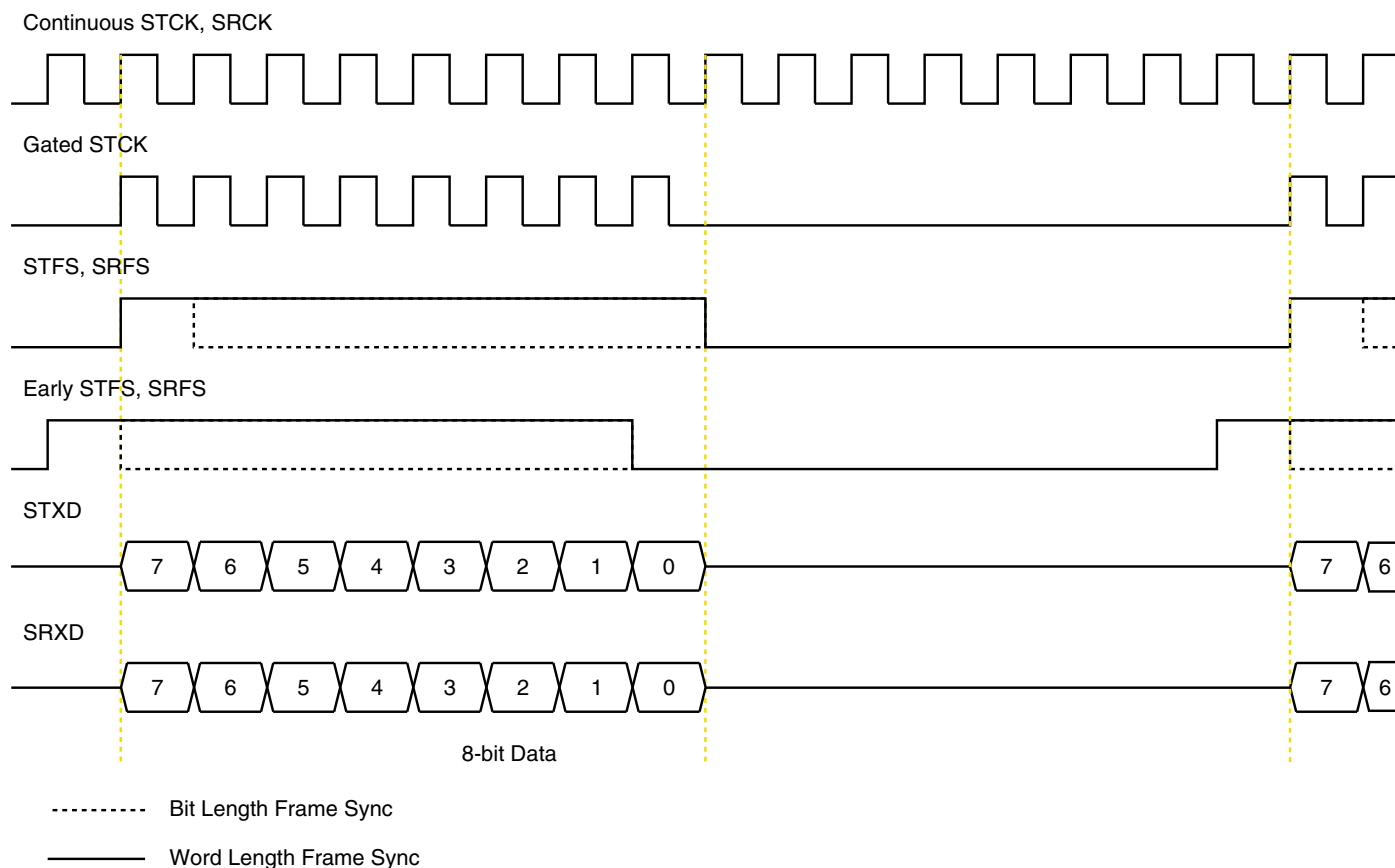


SSI Internal Continuous Clock for TX (RXDIR = 0, TXDIR = 1, RFDIR = 0, TFDIR = 1, SYN = 0)  
SSI External Continuous Clock for RX

**Figure 48-2. Asynchronous (SYN=0) SSI Configurations-Continuous Clock**

## External Signal Description

See the following figure for an example of the port signals for an 8-bit data transfer. Continuous and gated clock signals are shown, as well as the bit-length frame sync signal and the word-length frame sync signal.



**Figure 48-3. Serial Clock and Frame Sync Timing**

See the table below for list of clock pin configurations.

**Table 48-2. Clock Pin Configurations**

SYN	RXDIR	TXDIR	RFDIR	TFDIR	SRCK	STCK	SRFS	STFS
Asynchronous Mode								
0	0	0	0	0	RCK in	TCK in	RFS in	TFS in
0	0	0	0	1	RCK in	TCK in	RFS in	TFS out
0	0	0	1	0	RCK in	TCK in	RFS out	TFS in
0	0	0	1	1	RCK in	TCK in	RFS out	TFS out
0	0	1	0	0	RCK in	TCK out	RFS in	TFS in
0	0	1	0	1	RCK in	TCK out	RFS in	TFS out
0	0	1	1	0	RCK in	TCK out	RFS out	TFS in
0	0	1	1	1	RCK in	TCK out	RFS out	TFS out
0	1	0	0	0	RCK out	TCK in	RFS in	TFS in

Table continues on the next page...

**Table 48-2. Clock Pin Configurations (continued)**

SYN	RXDIR	TXDIR	RFDIR	TFDIR	SRCK	STCK	SRFS	STFS
0	1	0	0	1	RCK out	TCK in	RFS in	TFS out
0	1	0	1	0	RCK out	TCK in	RFS out	TFS in
0	1	0	1	1	RCK out	TCK in	RFS out	TFS out
0	1	1	0	0	RCK out	TCK out	RFS in	TFS in
0	1	1	0	1	RCK out	TCK out	RFS in	TFS out
0	1	1	1	0	RCK out	TCK out	RFS out	TFS in
0	1	1	1	1	RCK out	TCK out	RFS out	TFS out
Synchronous Mode								
1	0	0	x	0	-	CK in	-	FS in
1	0	0	x	1	-	CK in	-	FS out
1	0	1	x	0	-	CK out	-	FS in
1	0	1	x	1	-	CK out	-	FS out
1	1	0	x	x	-	Gated in	-	-
1	1	1	x	x	-	Gated out	-	-

## 48.3 Clocks

The table found here describes the clock sources for SSI.

Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 48-3. SSI Clocks**

Clock name	Clock Root	Description
ccm_ssi_clk	ssi_clk_root	Module / system clock for bit clock generation
ipg_clk	ipg_clk_root	Peripheral clock
ipg_clk_s	ipg_clk_root	Peripheral access clock

## 48.4 SSI Transmit FIFO 0 & 1 Registers

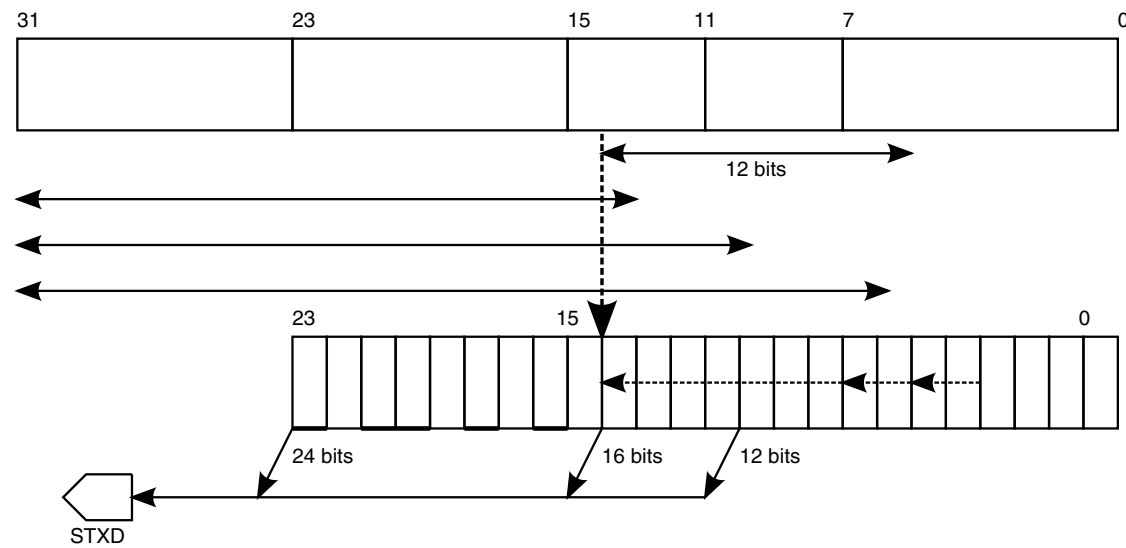
The SSI Transmit FIFO registers are 15x32-bit registers. These registers are not directly accessible by the end user. Transmit Shift Register (TXSR) receives its values from these FIFO registers. Transmitted data is first-in-first-out.

When the Transmit Interrupt Enable (TIE) bit in the SIER and either of the Transmit FIFO Empty Enable (TFE0 or 1) bits in the SIER are set, an interrupt is asserted whenever the number of empty slots exceed or are equal to the selected threshold value of corresponding Tx-FIFO. The threshold value is contained in the corresponding Transmit FIFO Watermark (TFWM0 or 1) field in the SSI FIFO Control/Status Register (SFCSR).

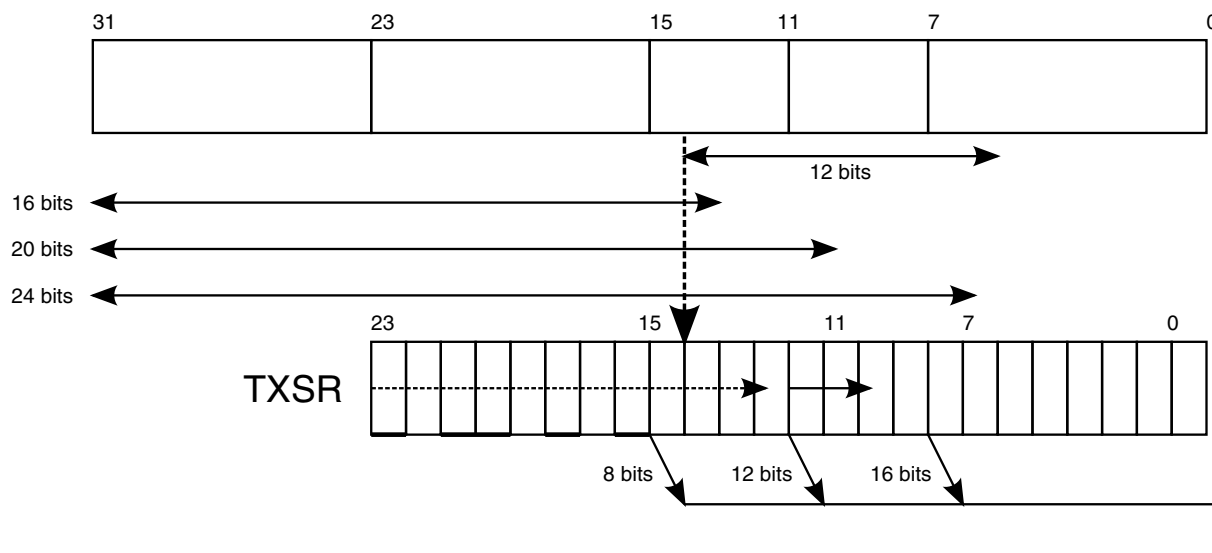
## **48.5 SSI Transmit Shift Register (TXSR)**

The SSI Transmit Shift Register (TXSR) is a 24-bit shift register that contains the data being transmitted.

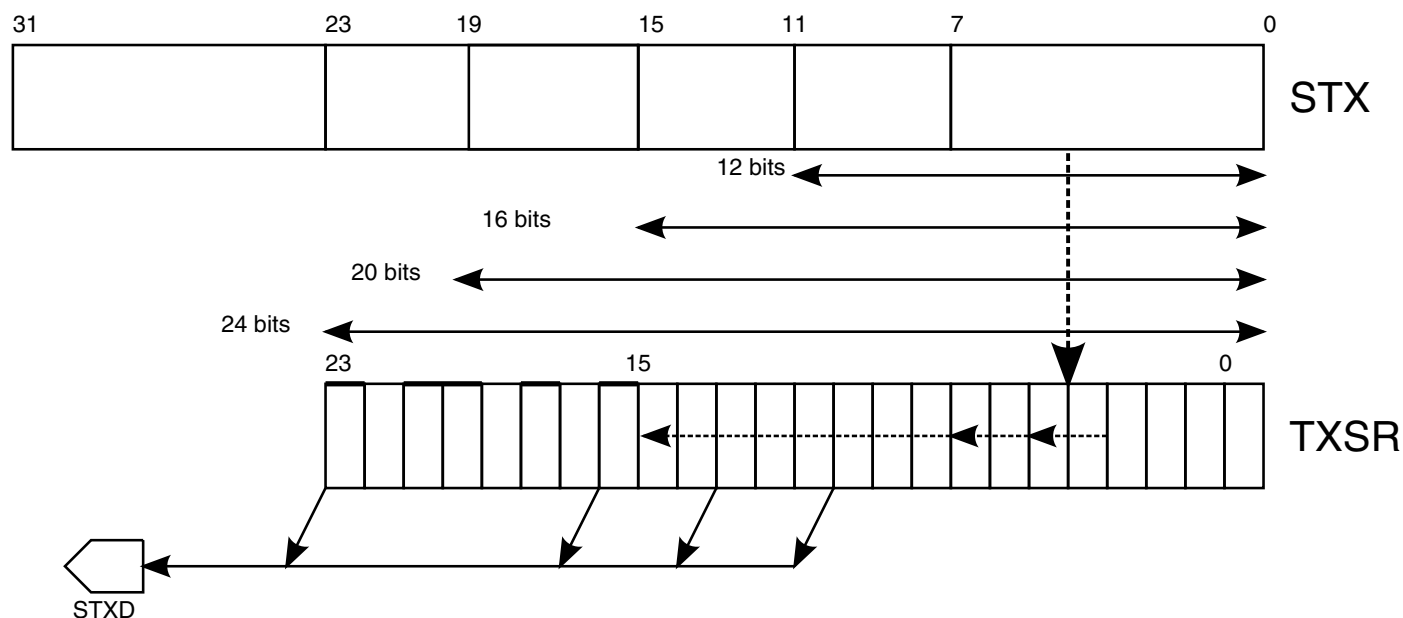
This register is not directly accessible by the end user. When a continuous clock is used, data is shifted out to the Serial Transmit Data (STXD) port by the selected (internal/external) bit clock when the associated (internal/external) frame sync is asserted. When a gated clock is used, data is shifted out to the STXD port by the selected (internal/external) gated clock. The Word Length control bits (WL[3:0]) in the SSI Transmit and Receive Clock Control Register (STCCR) determine the number of bits to be shifted out of the TXSR before it is considered empty and can be written to again. This word length can be 8, 10, 12, 16, 18, 20, 22 or 24 bits. The data to be transmitted occupies the most significant portion of the shift register if TXBIT0 is '0', otherwise it occupies the least significant portion. The unused portion of the register is ignored. Data is always shifted out of this register with the Most Significant Bit (MSB) first when the SHFD bit of the STCR is cleared. If this bit is set, the Least Significant Bit (LSB) is shifted out first. The figures below show the transmitter loading and shifting operation. The figures show the working for some WL values, the same can be extended for other values.



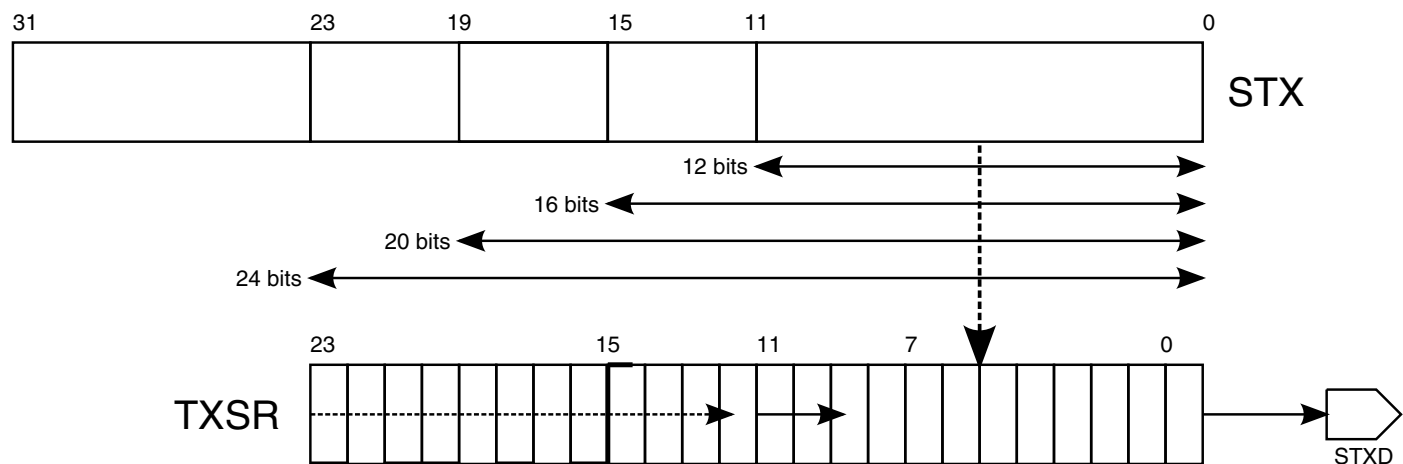
**Figure 48-4. Transmit Data Path (TXBIT0=0, TSHFD=0) (MSB Alignment)**



**Figure 48-5. Transmit Data Path (TXBIT0=0, TSHFD=1) (MSB Alignment)**



**Figure 48-6. Transmit Data Path (TXBIT0=1, TSHFD=0) (LSB Alignment)**



**Figure 48-7. Transmit Data Path (TXBIT0=1, TSHFD=1) (LSB Alignment)**

## 48.6 SSI Receive FIFO 0 and 1 Registers

The SSI Receive FIFO registers are 15x32-bit registers. These registers are not directly accessible by the end user. These FIFO registers receive data from the Receive Shift Register (RXSR). Received data is first-in-first-out.

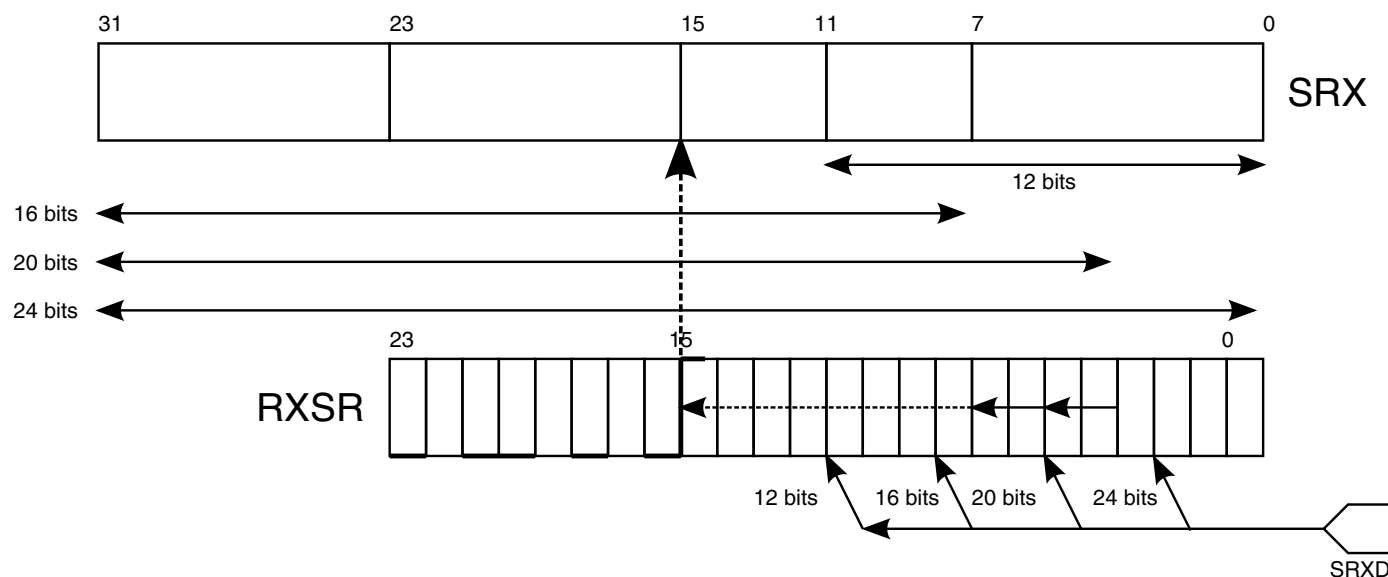


When the Receive Interrupt Enable (RIE) bit in the SIER and either of the Receive FIFO Full Enable (RFF0\_EN or RFF1\_EN) bits in the SIER are set, an interrupt is asserted whenever the number of full slots exceeds or is equal to the selected threshold value of corresponding Rx-FIFO. The threshold value is contained in the corresponding Receive FIFO Watermark (RFBM0 or 1) field in the SSI FIFO Control/Status Register (SFCSR).

## 48.7 SSI Receive Shift Register (RXSR)

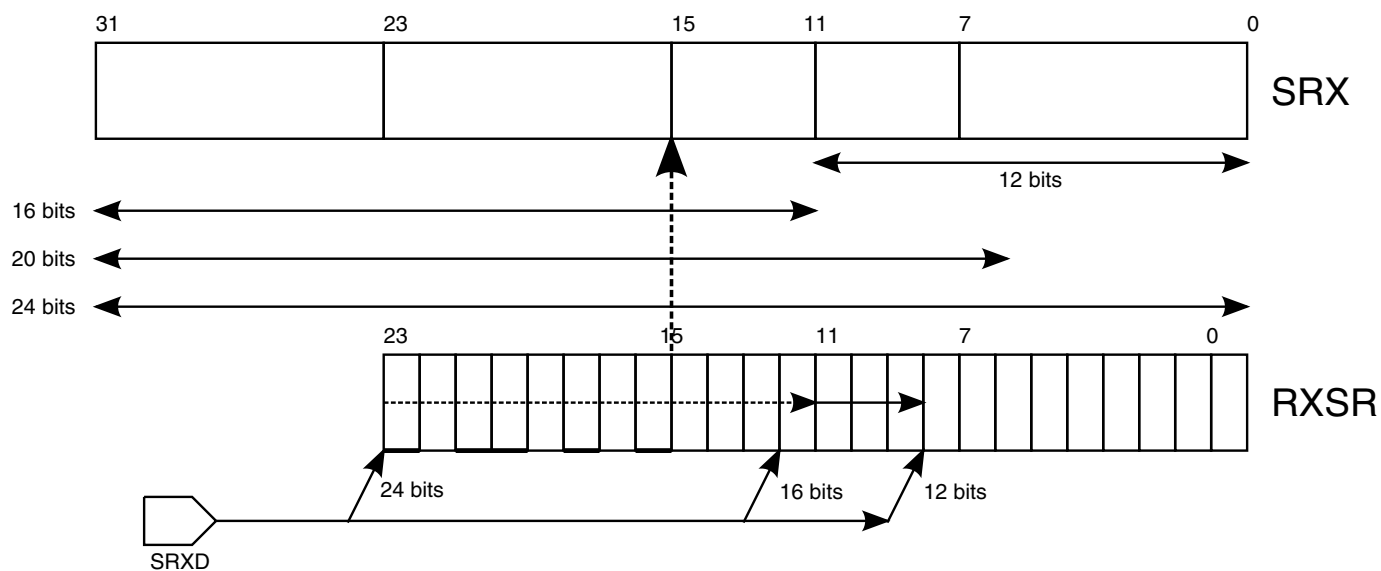
The SSI Receive Shift Register (RXSR) is a 24-bit, shift register that receives incoming data from the serial receive data SRXD port.

This register is not directly accessible by the end user. When a continuous clock is used, data is shifted in by the selected (internal/external) bit clock when the associated (internal/external) frame sync is asserted. When a gated clock is used, data is shifted in by the selected (internal/external) gated clock. Data is assumed to be received MSB first if the SHFD bit of the SSI.SRCR is cleared. If this bit is set, the data is received LSB first. Data is transferred to the appropriate SSI Receive Data Register (SRX0/1) or Receive FIFOs (if the receive FIFO is enabled and the corresponding SRX is full) after 8, 10, 12, 16, 18, 20, 22 or 24 bits have been shifted in depending on the WL[3:0] control bits. For receiving less than 24 bits of data, LSB bits are appended with zero. The figures below show the receiver loading and shifting operation. These figures show the operation for several values of WL and the same can be extended for other values.

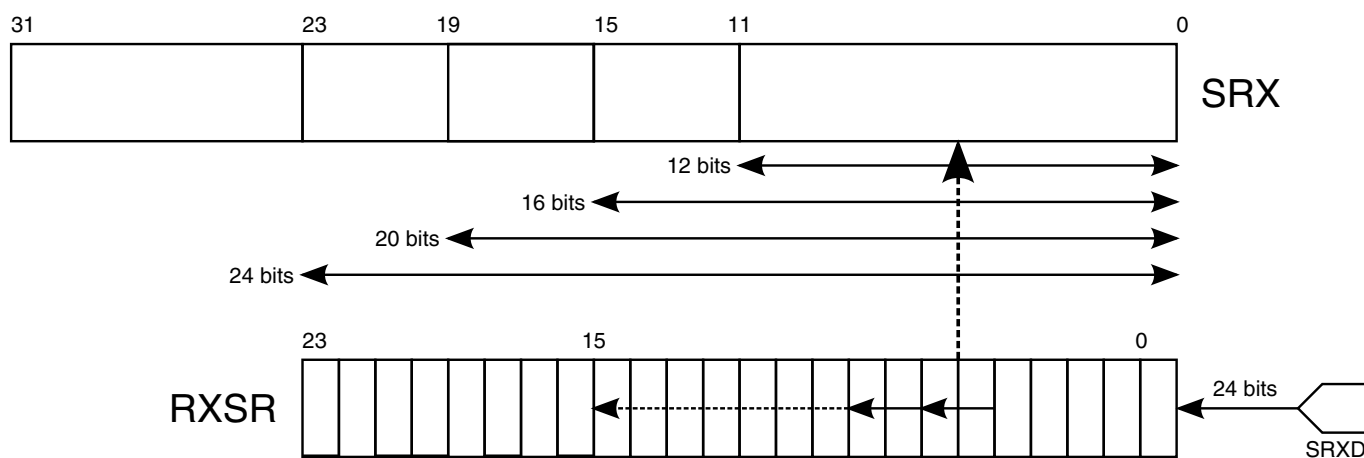


**Figure 48-8. Receive Data Path (RXBIT0=0, RSHFD=0) (MSB Alignment)**

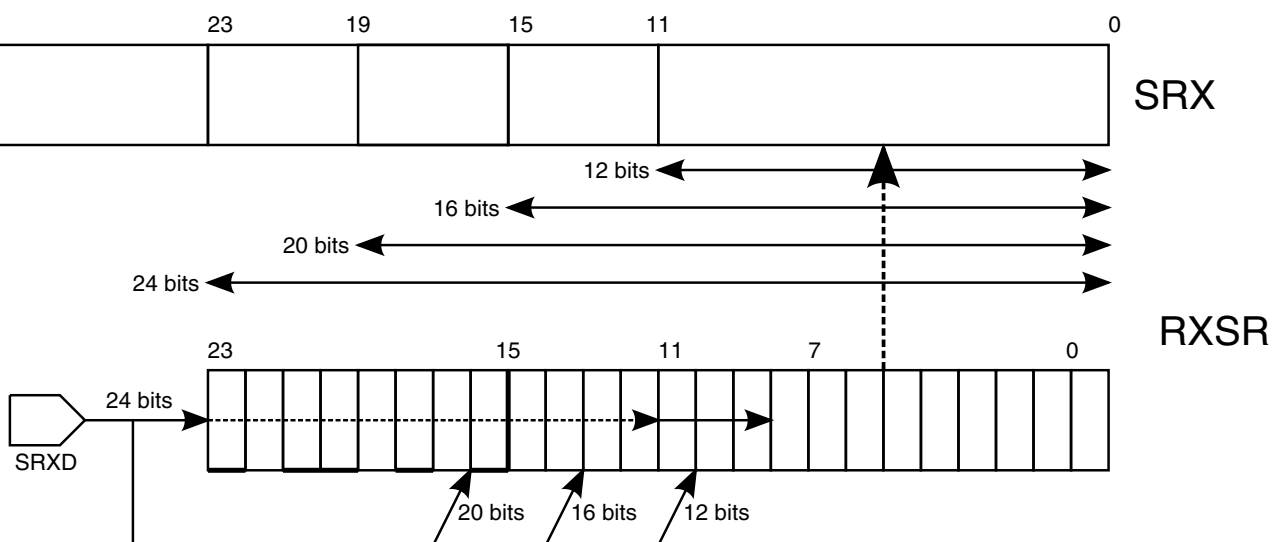
## SSI Receive Shift Register (RXSR)



**Figure 48-9. Receive Data Path (RXBIT0=0, RSHFD=1) (MSB Alignment)**



**Figure 48-10. Receive Data Path (RXBIT0=1, RSHFD=0) (LSB Alignment)**



**Figure 48-11. Receive Data Path (RXBIT0=1, RSHFD=1) (LSB Alignment)**

## 48.8 Functional Description

This section provides a complete functional description of the block.

### 48.8.1 Operating Modes

Different modes can be programmed by several bits in the SSI control registers.

See the table below for the list of SSI operating modes and some of the typical applications in which they can be used:

### Table 48-4. SSI Operating Modes

TX, RX Sections	Serial Clock	Mode	Typical Application
Asynchronous	Continuous	Normal	Multiple synchronous CODECs
Asynchronous	Continuous	Network	TDM CODEC or DSP networks
Synchronous	Continuous	Normal	Multiple synchronous CODECs
Synchronous	Continuous	Network	TDM CODEC or DSP network
Synchronous	Gated	Normal	SPI-type devices; DSP to ARM platform

The transmit and receive sections of the SSI can be synchronous or asynchronous. In Synchronous mode, the transmitter and the receiver use a common clock and frame synchronization signal. Masking of slots for Transmit and Receive section can differ in

synchronous mode. Also the shifting of data is independent and for receive section depends on RXBIT0 and RSHFD bits in SSI.SRCR register. In Asynchronous mode, the transmitter and receiver each has its own clock and frame synchronization signals.

Normal or Network mode can be selected. In Normal mode, the SSI functions with one data word of I/O per frame. In Network mode, any number from two to thirty-two data words of I/O per frame can be used. Network mode is typically used in star time division multiplex networks with other processors or CODECs, allowing interface to time division multiplexed networks without additional logic. Gated clock mode option can be selected in Normal synchronous mode only. During Gated mode the clock is not-continuous and runs only during data-transmission. These distinctions result in the basic operating modes that allow the SSI to communicate with a wide variety of devices.

The SSI supports both Normal and Network modes, and these can be selected independently of whether the transmitter and receiver are synchronous or asynchronous. Typically these protocols are used in a periodic manner, where data is transferred at regular intervals, such as at the sampling rate of an external CODEC. Both modes use the concept of a frame. The beginning of the frame is marked with a frame sync when programmed with continuous clock. The frame sync occurs at a periodic interval. The length of the frame is determined by the DC[4:0] bits in either the SSI.SRCCR or SSI.STCCR register, depending on whether data is being transmitted or received. The number of words transferred per frame depends on the mode of the SSI.

In Normal mode, one data word is transferred per frame. In Network mode, the frame is divided into anywhere between two and thirty-two time slots, where in each time slot one data word can optionally be transferred.

Apart from the above basic modes of operation, SSI supports the following modes which require some specific programming.

- I2S mode
- AC97 mode
  - AC97 Fixed mode
  - AC97 Variable mode

In (non-I2S) slave modes (external frame sync), the programmed word length setting of the SSI should be equal to the word length setting of the master. In I2S slave mode, the programmed word length setting of the SSI can be lesser than or equal to the word length setting of the I2S master (external CODEC).

In slave modes, the programmed frame length setting (DC bits) of the SSI can be lesser than or equal to the frame length setting of the master (external CODEC).

The following sections provide detailed descriptions of the above modes.

### 48.8.1.1 Normal Mode

Normal mode is the simplest mode of the SSI. It is used to transfer data in one time slot per frame.

A time slot is a unit of data and the WL[3:0] bits define the number of bits in a time slot. In Continuous Clock mode, a frame sync occurs at the beginning of each frame. The length of the frame is determined by the following factors:

- The period of the Serial Bit Clock (DIV2, PSR, PM[7:0] bits for internal clock or the frequency of the external clock on the STCK port)
- The number of bits per time slot (WL[3:0] bits)
- The number of time slots per frame (DC[4:0] bits)

If Normal mode is configured with more than one time slot per frame, data is transferred only in the first time slot. No data is transferred in subsequent time slots. In Normal mode, DC[4:0] values corresponding to more than a single time slot in a frame, only results in lengthening of the frame.

#### 48.8.1.1.1 Normal Mode Transmit

The conditions for data transmission from the SSI in Normal mode are:

- SSI Enabled (SSIEN = 1)
- Write data to Transmit Data Register (STX)
- Transmitter Enabled (TE = 1)
- Frame sync active (for continuous clock case)
- Bit clock begins

When the above conditions occur in Normal mode, the next data word is transferred into the Transmit Shift Register (SSI.TXSR) from the Transmit Data Register 0 (SSI.STX0), or from the Transmit FIFO 0 Register, if transmit FIFO 0 is enabled. The new data word is transmitted on arrival of frame-sync preceded by clocks in continuous clock mode. In gated-external mode, data word is transmitted on arrival of frame-sync. In gated-internal mode, data word is transmitted whenever data is available in Tx-FIFO.

If Transmit FIFO 0 is not enabled and the transmit data register empty enable (TDE0\_EN) and transmit interrupt enable (TIE) bits are set, transmit interrupt occurs when the word in SSI\_STX0 is shifted to Transmit Shift (SSI.TXSR) register for shifting.

If Transmit FIFO 0 is enabled and the transmit fifo empty enable (TFE0\_EN) and transmit interrupt enable (TIE) bits are set, transmit interrupt occurs when the number of empty slots in Transmit Fifo 0 exceed or are equal to the selected threshold value i.e.

Transmit Fifo 0 Watermark (TFWM0) value. If transmit FIFO 0 is enabled and filled with data, 15 data words can be transferred before the core must write new data to the SSI.STX0 register.

The STXD port is disabled except during the data transmission period. For a continuous clock, the optional frame sync output and clock outputs are not disabled, even if both receiver and transmitter are disabled.

#### **48.8.1.1.2 Normal Mode Receive**

The conditions for data reception from the SSI are:

- SSI enabled (SSIEN = 1)
- Receiver enabled (RE = 1)
- Frame sync active (for continuous clock case)
- Bit clock begins

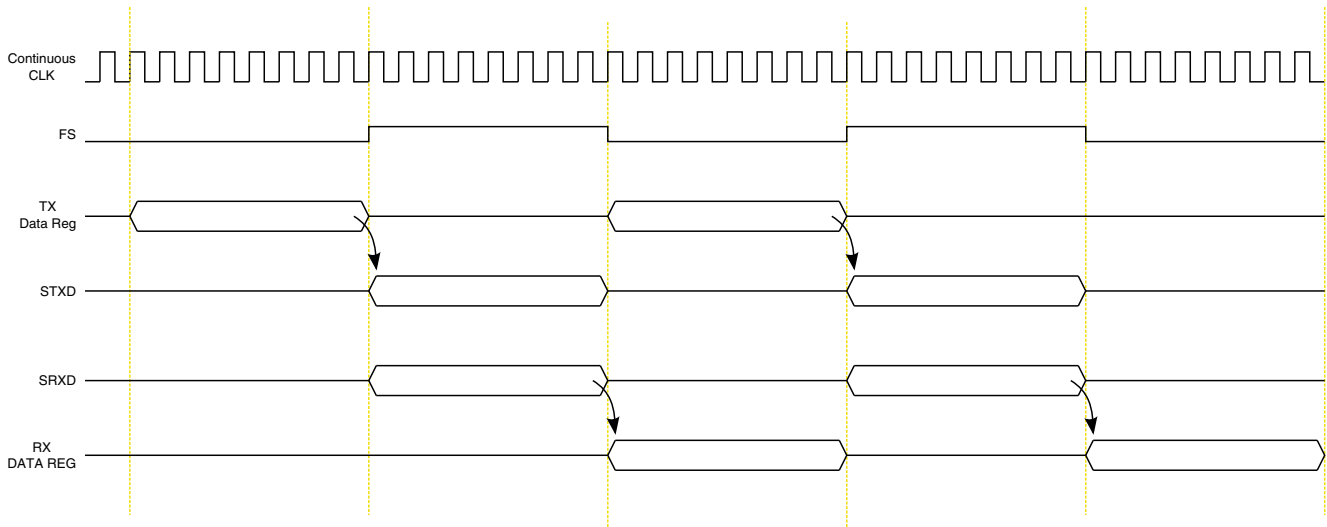
With the above conditions in Normal mode with a continuous clock, each time the frame sync signal is generated (or detected) a data word is clocked in. With the above conditions and a gated clock, each time the clock begins, a data word is clocked in.

If Receive FIFO 0 is not enabled and Receive Interrupt enable (RIE) and Received Data 0 Ready enable (RDR0\_EN) bits are set, receive interrupt occurs when received data word is transferred from the Receive Shift Register (SSI.RXSR) to the Receive Data Register 0 (SSI.SRX0), thus setting the Receive Data Ready 0 (RDR0) flag.

If Receive FIFO 0 is enabled, and Receive Interrupt enable (RIE) and Received Fifo 0 full enable (RFF0\_EN) bits are set, receive interrupt occurs when the received data word is transferred to the Receive FIFO 0 and Receive FIFO 0 reaches the selected threshold and results in Receive FIFO Full 0 (RFF0) flag to get set.

The core program has to read the data from the Receive Data Register 0 (SSI.SRX0) (in case Receive FIFO0 is disabled) before a new data word is transferred from the Receive Shift Register (SSI.RXSR), otherwise the Receive Overrun Error 0 (ROE0) bit is set. If receive FIFO 0 is enabled, the Receive Overrun Error 0 (ROE0) bit is set when the Receive FIFO 0 data level reaches the selected threshold and a new data word is ready to be transferred to the Receive FIFO 0.

See the following figure for an illustration of transmitter and receiver timing for an 8-bit word in the first time slot in Normal mode, continuous clock with a late word length frame sync. The Tx Data register is loaded with the data to be transmitted. On arrival of the clock, this data is transferred to the Transmit Shift Register, which gets transmitted on arrival of the frame-sync on the STXD output. Simultaneously, the Receive Shift Register shifts in the received data available on the SRXD input and, at the end of the time slot, this data is transferred to the Rx Data Register.



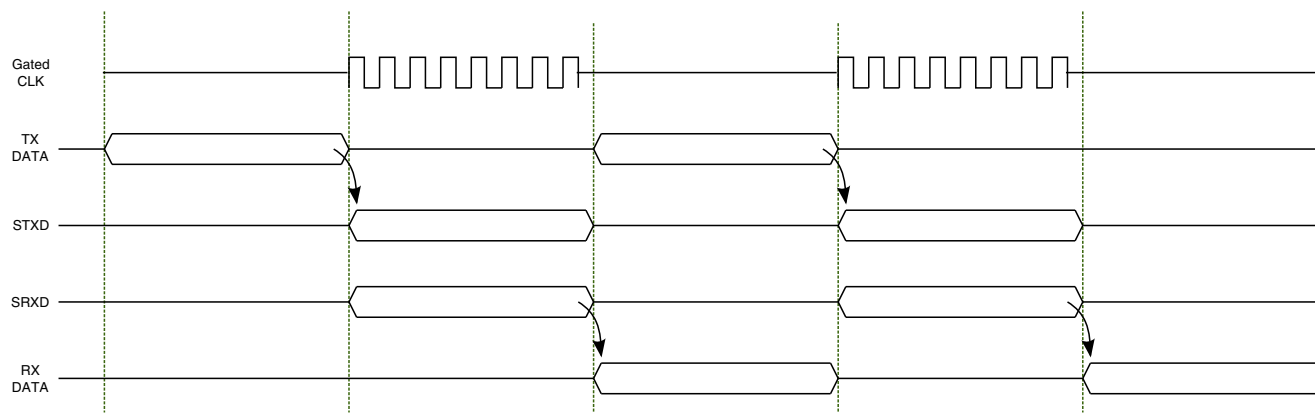
**Figure 48-12. Normal Mode Timing - Continuous Clock**

The following figure shows a similar case for internal (SSI generates clock) gated clock mode and [Figure 48-14](#) shows a case for external (SSI receives clock) gated clock mode.

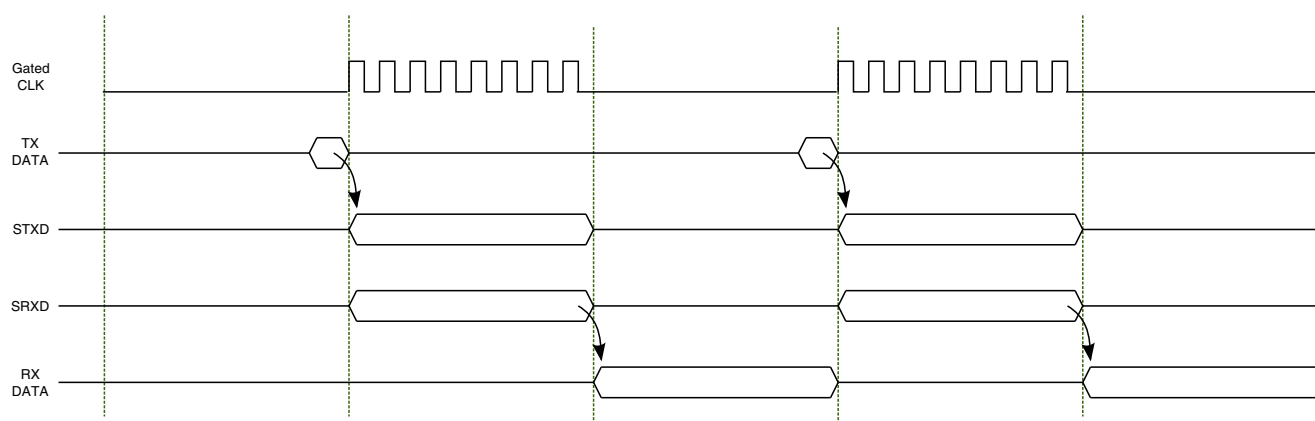
### NOTE

A pull-down resistor is required in the gated clock case because the clock port is disabled between transmissions.

The Tx Data register is loaded with the data to be transmitted. On arrival of the clock, this data is transferred to the Transmit Shift Register, which gets transmitted on arrival of the frame-sync on the STXD output. Simultaneously, the Receive Shift Register shifts in the received data available on the SRXD input and, at the end of the time slot, this data is transferred to the Rx Data Register. In case of Internal Gated clock mode, the Tx Data line and clock output port are put in the high-impedance state at the end of transmission of the last bit (at the completion of the complete clock cycle), whereas, in External Gated clock mode, the Tx Data line is tri-stated at the last inactive edge of the incoming bit clock (during the last bit in a data word).



**Figure 48-13. Normal Mode Timing - Internal Gated Clock**



**Figure 48-14. Normal Mode Timing - External Gated Clock**

## 48.8.1.2 Network Mode

Network mode is used for creating a Time Division Multiplexed (TDM) network, such as a TDM CODEC network or a network of DSPs.

In Continuous Clock mode, a frame sync occurs at the beginning of each frame. In this mode, the frame is divided into more than one time slot. During each time slot, one data word can be transferred. Each time slot is then assigned to an appropriate CODEC or DSP on the network. The DSP can be a master device that controls its own private network, or a slave device that is connected to an existing TDM network and occupies a few time slots.



The frame sync signal indicates the beginning of a new data frame. Each data frame is divided into time slots and transmission and/or reception of one data word can occur in each time slot (rather than in just the frame sync time slot as in Normal mode). The frame rate dividers, controlled by the DC[4:0] bits, select two to thirty-two time slots per frame. The length of the frame is determined by the following factors:

- The period of the serial bit clock (PSR, PM[7:0] bits for internal clock, or the frequency of the external clock on the STCK port)
- The number of bits per sample (WL[3:0] bits)
- The number of time slots per frame (DC[4:0] bits)

In Network mode, data can be transmitted in any time slot. The distinction of the Network mode is that each time slot is identified with respect to the frame sync (data word time). This time slot identification allows the option of transmitting data during the time slot by writing to the STX registers or ignoring the time slot as determined by SSI.STMSK register bits. The receiver is treated in the same manner and received data is only transferred to the receive data register/fifo if the corresponding time slot is enabled (through SSI.SRMSK).

By utilizing the SSI.STMSK and SSI.SRMSK registers, software only has to service the SSI during valid time slots. This eliminates any overhead associated with unused time slots. See [SSI Memory Map/Register Definition](#) for more information on SSI.STMSK and SSI.SRMSK.

In the Two-Channel mode of operation, the second set of Transmit and Receive FIFOs and Data Registers are used to create two separate channels. These channels are completely independent, with a their own set of Core interrupts and DMA requests, which are identical to the ones available for the default channel. In this mode, data is transmitted/received in enabled time slots alternately from/to FIFO 0 and FIFO 1, starting from FIFO 0. The first data word is taken from FIFO 0 and transmitted in the first enabled time slot and subsequently, data is loaded from FIFO 1 and FIFO 0 alternately and transmitted. Similarly, the first received data is sent to FIFO 0 and subsequent data is sent to FIFO 1 and FIFO 0 alternately. Time slots can be selected through the Transmit and Receive Time Slot Mask registers (SSI.STMSK and SSI.SRMSK). For using this mode of operation, the TCH\_EN bit (SCR[8]) needs to be set.

#### 48.8.1.2.1 Network Mode Transmit

The transmit portion of SSI is enabled when the SSIEN and the TE bits in the SSI.SCR are both set. However, for continuous clock, when the TE bit is set, the transmitter is enabled only after detection of a new frame sync (transmission starts from the next frame boundary).

Normal start-up sequence for transmission is to perform the following:

1. Enable Network Mode.
2. Enable SSI
3. Write the data to be transmitted to the SSI.STX register. This clears the TDE flag
4. Set the TE bit to enable the transmitter on the next frame boundary (for continuous clock case).
5. Enable transmit interrupts.

(Alternatively, the programmer may decide not to transmit in a time slot by configuring the STMSK.) TDE flag is set as data is shifted from SSI.STX register to TXSR, but the STXD port remains disabled during the time slots. When the next frame sync is detected or generated (continuous clock), the data word in TXSR and is shifted out (transmitted). When the SSI.STX register is empty, the TDE bit is set, which causes a transmitter interrupt (in case the FIFO is disabled) to be sent if the TIE bit is set. Software can poll the TDE bit or use interrupts to reload the STX register with new data for the next time slot. Failing to reload the SSI.STX register before the TXSR is finished shifting (empty) causes a transmitter underrun and the TUE error bit is set. In case the FIFO is enabled, the TFE flag is set in accordance with the watermark setting and this flag causes the transmitter interrupt to occur.

The operation of clearing the TE bit disables the transmitter after completion of transmission of the current frame. Setting the TE bit enables transmission from the next frame. During that time the STXD port is disabled. The TE bit should be cleared after the TDE bit is set to ensure that all pending data is transmitted.

To summarize, the Network mode transmitter generates interrupts every enabled time slot (when FIFO is disabled) and requires the core program to respond to each enabled time slot. These responses from the core are one of the following:

- Write data in data register to enable transmission in the next time slot.
- Configure the time slot register to disable transmission in the next time slot (unless time slot is already masked by SSI.STMSK register bit).
- Do nothing-transmit underrun occurs at the beginning of the next time slot and the previous data is re-transmitted.

In the Two-Channel mode of operation, both the channels (Data Registers, FIFOs, Interrupts and DMA requests) operate in the same manner, as described above. The only difference in case of the second channel is that the Interrupts related to this channel are generated only in case this mode of operation is selected (TDE1 is low by default).

### 48.8.1.2.2 Network Mode Receive

The receiver portion of the SSI is enabled when both the SSIEN and the RE bits in the SSI.SCR are set. However, the receive enable only takes place during that time slot if RE is enabled before the second to last bit of the word. If the RE bit is cleared, the receiver is disabled at the end of the current frame.

SSI is capable of finding the start of the next frame automatically. When the word is completely received, it is transferred to the SRX register, which sets the RDR bit (Receive Data Ready). Setting the RDR bit causes a receive interrupt to occur if the receiver interrupt is enabled (the RIE bit is set) and (Receive data ready enable) RDR\_EN bit is set. The second data word (second time slot in the frame), begins shifting in immediately after the transfer of the first data word to the SSI.SRX register. The core program has to read the data from the Receive Data Register (which clears RDR) before the second data word is completely received (ready to transfer to RX data register) or a receive overrun error occurs (the ROE bit is set).

An interrupt can occur after the reception of each enabled data word or the programmer can poll the RDR flag. The core program response can be one of the following:

- Read RX and use the data.
- Read RX and ignore the data.
- Do nothing-the receiver overrun exception occurs at the end of the current time slot.

#### NOTE

For a continuous clock, the optional frame sync output and clock output signals are not affected, even if the transmitter or receiver is disabled. TE and RE do not disable the bit clock or the frame sync generation. To disable the bit clock and the frame sync generation, the SSIEN bit in the SSI.SCR can be cleared or TFR\_CLK\_DIS/RFR\_CLK\_DIS bits can be set, or the port control logic external to the SSI (for example, in the IOMUXC) can be re configured.

In the Two-Channel mode of operation, both the channels (Data Registers, FIFOs, Interrupts and DMA requests) operate in the same manner, as described above. The only difference in case of the second channel is that the Interrupts related to this channel are generated only in case this mode of operation is selected.

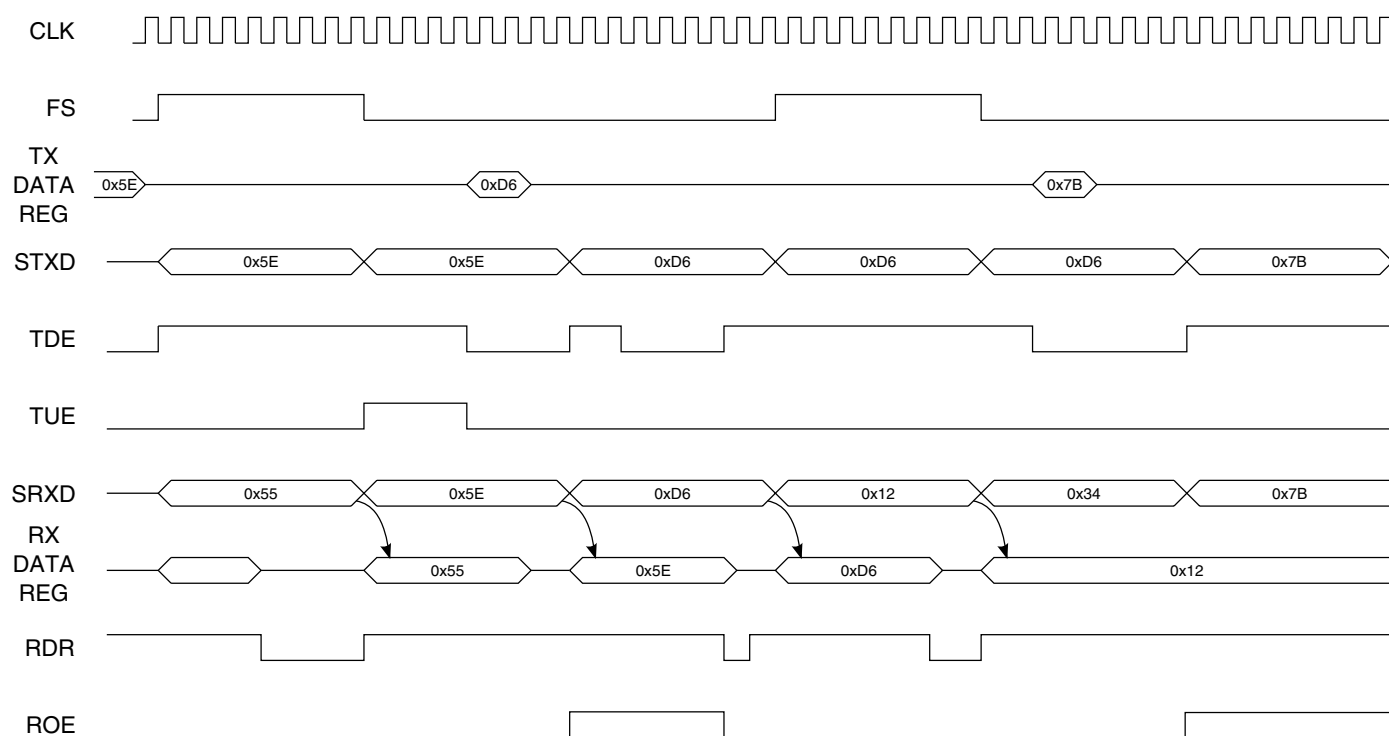
The transmitter and receiver timing for an 8-bit word with continuous clock, FIFO disabled, three words per frame sync in Network mode is shown in the figure below).

#### NOTE

The transmitter repeats the value 0x5E because of an underrun condition

## Functional Description

For the receive section, data received on the SRXD pin gets transferred to the Rx Data register at the end of each time slot. If the FIFO is disabled, the RDR flag gets set and causes a receiver interrupt if RE, RIE and RDR\_EN bits are set. If the FIFO is enabled, then the RFF flag is used for interrupt generation (this flag is set in accordance with the watermark settings). Here all time slots are enabled. The receive data ready flag is set after reception of the first data (0x55). Since the flag is not cleared (Rx Data Register is not read by core), the Receive Overrun Error (ROE) flag is set on reception of the next data (0x5E). ROE flag is cleared on writing '1' to the corresponding interrupt status bit in SSI Status Register.



Note: Processor must write to '1' to the corresponding TUE/ROE Interrupt status bit in SISR to clear TUE/ROE Interrupt.

**Figure 48-15. Network Mode Timing - Continuous Clock**

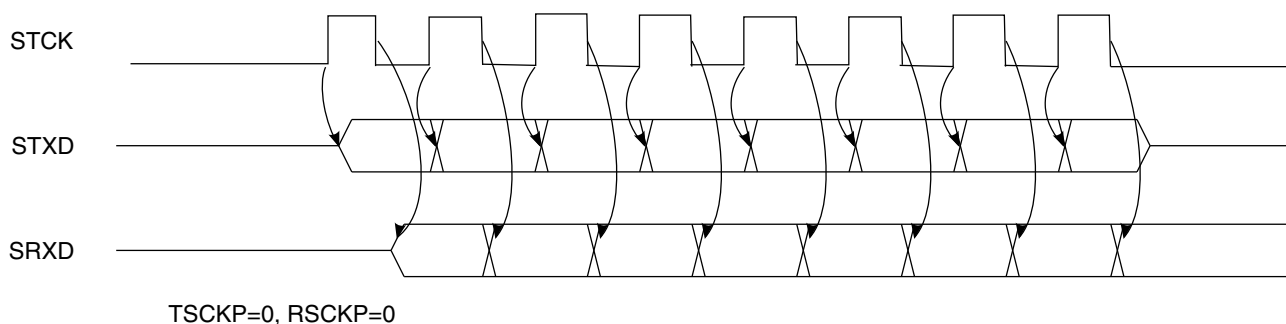
### 48.8.1.3 Gated Clock Mode

Gated Clock mode is often used to hook up to SPI-type interfaces on Micro controller Units (MCUs) or external peripheral chips. In Gated Clock mode, the presence of the clock indicates that valid data is on the STXD or SRXD ports.

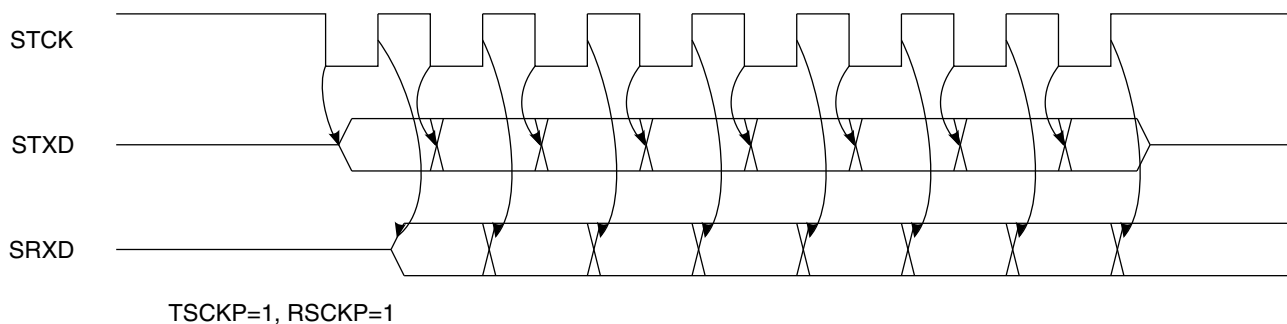
For this reason, no frame sync is needed in this mode. Once transmission of data has completed, the clock is pulled to the inactive state. Gated clocks are allowed for both the transmit and receive sections with either internal or external clock in Normal mode. Gated clocks are not allowed in Network mode. See [Table 48-2](#) ("Clock Pin Configurations") for SSI configuration for gated-mode operation.

The clock runs when the TE bit and/or the RE bit are appropriately enabled. For the case of internally generated clock, all internal bit clocks, word clocks, and frame clocks continue to operate. When a valid time slot occurs (such as the first time slot in Normal mode), the internal bit clock is enabled onto the appropriate clock port. This allows data to be transferred out in periodic intervals in Gated Clock mode. With an external clock, the SSI waits for a clock signal to be received. Once the clock begins, valid data is shifted in. Care should be taken to clear all DC bits (0x00000) when SSI is used in Gated mode. In gated mode of operation the TFS, RFS, TLS, RLS, TFRC and RFRC bits of SSI.AISR register are not generated.

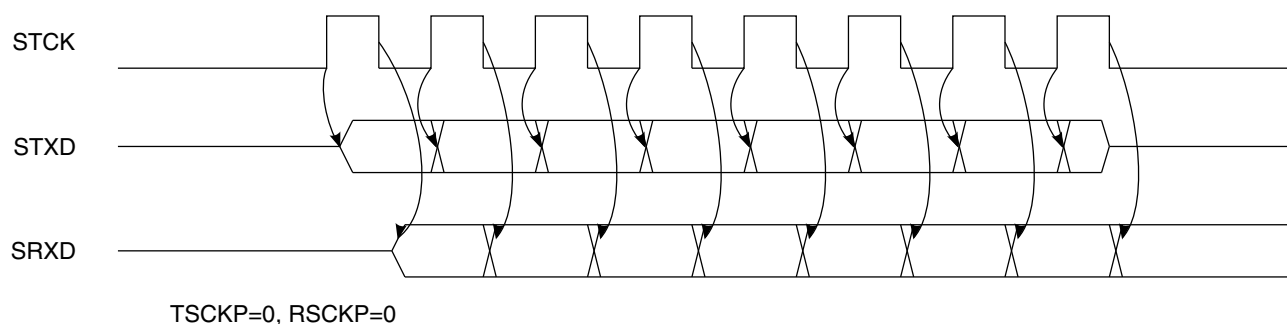
For Gated clock operated in external clock mode, a proper clock signalling must be applied to the SSI STCK in order for it to function properly. When TSCKP is 0, CLK\_IST value should be 1. When TSCKP is 1, CLK\_IST value should be 0. If the SSI uses rising edge transition to clock data (TSCKP=0) and the falling edge transition to latch data (RSCKP=0), the clock must be in an active low state when idle. If the SSI uses falling edge transition to clock data (TSCKP=1) and the rising edge transition to latch data (RSCKP=1), the clock must be in an active high state when idle. The figures below illustrate the different edge clocking/latching.



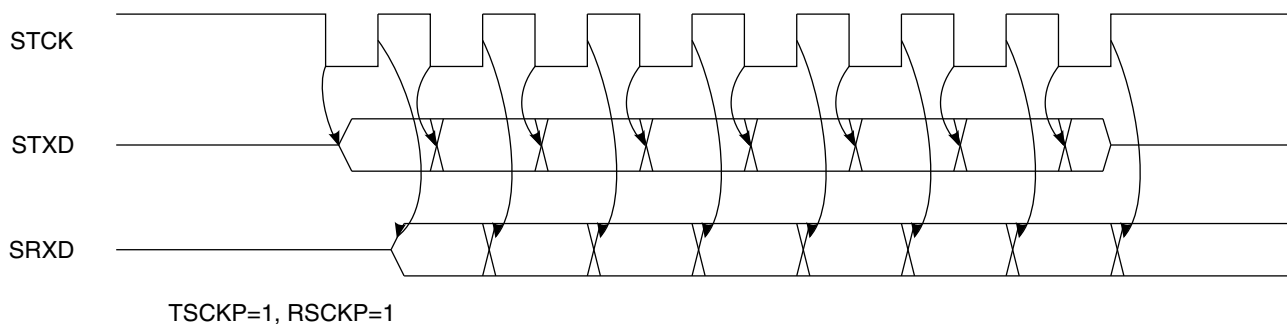
**Figure 48-16. Internal Gated Mode Timing - Rising Edge Clocking / Falling Edge Latching**



**Figure 48-17. Internal Gated Mode Timing - Falling Edge Clocking / Rising Edge Latching**



**Figure 48-18. External Gated Mode Timing - Rising Edge Clocking / Falling Edge Latching**



**Figure 48-19. External Gated Mode Timing - Falling Edge clocking / Rising Edge Latching**

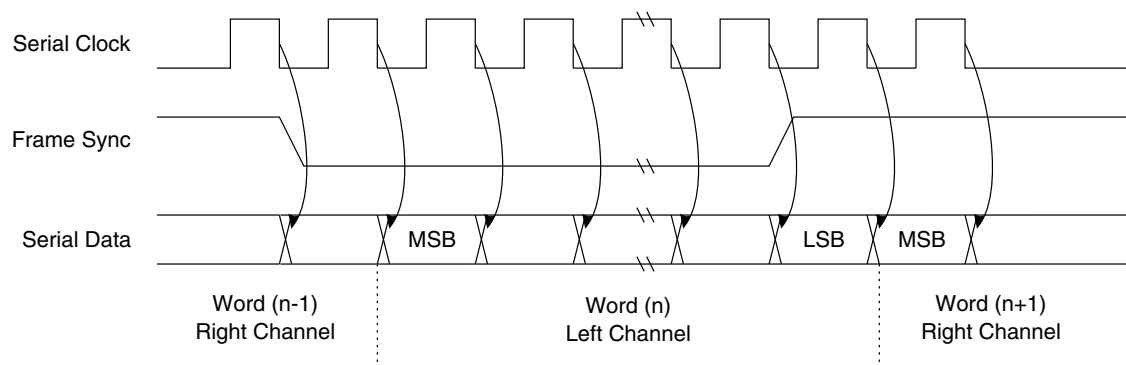
The bit clock ports must be kept free of timing glitches. If a single glitch occurs, all ensuing transfers will be out of synchronization.

In case of External Gated Mode, even though the Tx Data line is put in the high-impedance state at the last non-active edge of the bit clock, the round trip delay should be sufficient to take care of hold time requirements at the external receiver.

### 48.8.1.4 I2S Mode

The SSI is compliant to the Inter-IC Sound (I2S) bus specification from Philips Semiconductors (February 1986, Revised June 5, 1996). For more information on I2S, refer to the latest version of NXP Semiconductor's I2S specification.

See the following figure for an illustration of the basic I2S protocol timing.



**Figure 48-20. I2S Mode Timing - Serial Clock, Frame Sync and Serial Data**

Select I2S mode using the options listed in the table below.

**Table 48-5. I2S Mode Selection**

I2S_MODE[1]	I2S_MODE[0]	Mode Type
0	0	Normal mode
0	1	I2S master mode
1	0	I2S slave mode
1	1	Normal mode

In normal mode operation, no register bits are forced to any particular state internally and the user can program the SSI to work in any operating condition.

When I2S modes are entered (I2S master (01) or I2S slave (10)), the following settings are recommended:

- Sync mode (SSI\_SCR[4] = 1)
- Tx shift direction: MSB transmitted first (SSI\_STCR[4]=0)
- Rx shift direction: MSB received first (SSI\_SRCR[4]=0)
- Tx data clocked at falling edge of the clock (SSI\_STCR[3]=1)
- Rx data latched at rising edge of the clock (SSI\_SRCR[3]=1)
- Tx frame sync active low (SSI\_STCR[2]=1)
- Rx frame sync active low (SSI\_SRCR[2]=1)

## Functional Description

- Tx frame sync initiated one bit before data is transmitted (SSI\_STCR[0]=1)
- Rx frame sync initiated one bit before data is received (SSI\_SRCR[0]=1)
- TX Frame Rate should be 2 (SSI\_STCCR[12:8] = 1)
- RX Frame Rate should be 2 (SSI\_SRCR[12:8] = 1)

In I2S master mode (SSI\_SCR[6:5]=01), the following additional settings are recommended:

- TXDIR bit (SSI\_STCR[5]) set to 1 to select internal generated bit clock
- TFDIR bit (SSI\_STCR[6]) set to 1 to select internal generated frame sync

In I2S master mode (SSI\_SCR[6:5]=01), the following settings are internally overridden by the hardware:

- Network mode is selected (SSI\_SCR[3]=1)
- Tx frame sync length set to one-word-long-frame (SSI\_STCR[1]=0)
- Rx frame sync length set to one-word-long-frame (SSI\_SRCR[1]=0)
- Tx shifting w.r.t. bit 0 of TXSR (SSI\_STCR[9]=1)
- Rx shifting w.r.t. bit 0 of RXSR (SSI\_SRCR[9]=1)

The user needs to set the following control bits to configure the bit clock and frame sync:

- PM (SSI\_STCCR[7:0])
- PSR (SSI\_STCCR[17])
- DIV2 (SSI\_STCCR[18])
- WL (SSI\_STCCR[16:13])
- DC (SSI\_STCCR[12:8])

The word length is fixed to 32 in I2S Master mode and the WL bits determine the number of bits that will contain valid data (out of the 32 transmitted/received bits in each channel).

In I2S slave mode (SSI\_SCR[6:5]=10), the following additional settings are recommended:

- TXDIR bit (SSI\_STCR[5]) set to 0 to select external generated bit clock
- TFDIR bit (SSI\_STCR[6]) set to 0 to select external generated frame sync

In I2S slave mode (SSI\_SCR[6:5]=10), the following settings are internally overridden by the hardware:

- Normal mode is selected (SSI\_SCR[3]=0)
- Tx frame sync length set to one-bit-long-frame (SSI\_STCR[1]=1)
- Rx frame sync length set to one-bit-long-frame (SSI\_SRCR[1]=1)
- Tx shifting w.r.t. bit 0 of TXSR (SSI\_STCR[9]=1)
- Rx shifting w.r.t. bit 0 of RXSR (SSI\_SRCR[9]=1)



The user needs to set the following control bits to configure the data transmission:

- WL (SSI\_STCCR[16:13])
- DC (SSI\_STCCR[12:8])

The word length is variable in I2S slave mode and the WL bits determine the number of bits that will contain valid data. The actual word length is determined by the external CODEC. The external I2S Master still sends frame sync according to the I2S protocol (early, word wide and active low), the SSI internally operates so that each frame sync transition is the start of a new frame (the WL bits determine the number of bits to be transmitted/received). After one data word has been transferred, the SSI waits for the next frame sync transition to start operation in the next time slot. Transmit (STMSK) and receive (SRMSK) mask bits should not be used in I2S Slave mode of operation. Masking is supported only for network mode of operation.

#### 48.8.1.5 AC97 Mode

In AC97 mode of operation, the SSI transmits a 16-bit Tag Slot at the start of a frame and the rest of the slots (in that frame) are all 20-bits wide.

The same sequence is followed while receiving data. Refer to the AC97 specification for details regarding transmit and receive sequences and data formats.

#### NOTE

The Audio Codec specification released in 1997 [AC '97] defines the Architecture and Digital Interface, specifically designed for implementing audio and modem I/O functionality in personal computers. Companion specifications include the Modem Codec [MC '97], and the combined Audio/Modem Codec standard [AMC '97]. The current version of AC '97 was produced in 2002. The AC-97 specification defines a recommended 48-pin QFP IC package.

Note that the SSI only has one RxDATA pin so the SSI can only support one codec. Secondary codecs are not supported.

When AC97 mode is enabled, the following settings are internally overridden by the hardware. The programmed register values are not changed by entering AC97 mode but they no longer apply to the block's operation. Writing to the programmed register fields will update their values; these updates can be seen by reading back the register fields. However, these settings will not take effect until AC97 mode is turned off.

The register bits within the bracket are the equivalent settings:

- Sync mode is entered (SSI.SCR[4] =1)
- Network mode is selected (SSI.SCR[3]=1)
- Tx shift direction is MSB transmitted first (SSI.STCR[4]=0)
- Rx shift direction is MSB received first (SSI.SRCR[4]=0)
- Tx data is clocked at rising edge of the clock (SSI.STCR[3]=0)
- Rx data is latched at falling edge of the clock (SSI.SRCR[3]=0)
- Tx frame sync is active high (SSI.STCR[2]=0)
- Rx frame sync is active high (SSI.SRCR[2]=0)
- Tx frame sync length is one-word-long-frame (SSI.STCR[1]=0)
- Rx frame sync length is one-word-long-frame (SSI.SRCR[1]=0)
- Tx frame sync initiated one bit before data is transmitted (SSI.STCR[0]=1)
- Rx frame sync initiated one bit before data is received (SSI.SRCR[0]=1)
- Tx shifting w.r.t. bit 0 of TXSR (SSI.STCR[9]=1)
- Rx shifting w.r.t. bit 0 of RXSR (SSI.SRCR[9]=1)
- Tx FIFO is enabled (SSI.STCR[7]=1)
- Rx FIFO is enabled (SSI.SRCR[7]=1)
- TFDIR bit (SSI.STCR[6]) is forced to 1 internally to select internal generated frame sync
- TXDIR bit (SSI.STCR[5]) is forced to 0 internally to select external generated bit clock

Any alteration of these bits individually will not affect the operational conditions of the SSI unless AC97 mode is deselected.

Hence, the only control bits needed to be set by the user to configure the data transmission/reception are the WL (SSI.STCCR[16:13]) and DC (SSI.STCCR[12:8]) bits. In AC97 mode, the WL bits can only legally take the values corresponding to 16-bit (truncated data) or 20-bit time slots. In case WL bits are set to select 16-bit time slots, the SSI pads the transmit data (four least significant bits) with zeros and while receiving, stores only the most significant 16 bits in the Rx FIFO.

Follow the sequence for programming the SSI to work in AC97 mode:

1. Program the WL bits to a value corresponding to either 16 or 20 bits. The WL bit setting is only for the data portion of the AC97 frame (Slots #3 through #12). The Tag slot (Slot #0) is always 16 bits wide and the Command Address and Command Data slots (Slots #1 and #2) are always 20 bits wide.
2. Select the number of time slots by programming the DC bits. For AC97 operation, DC bits should be set to a value of '0xC', resulting in 13 time slots per frame.
3. Write data to be transmitted, in Tx FIFO 0 (through Tx Data Register 0) and Tx FIFO 1 while using Two-Channel Mode (TCH\_EN = 1).
4. Program the FV, TIF, RD, WR and FRDIV bits in SSI.SACNT register

5. Update the contents of SSI.SACADD, SSI.SACDAT and SSI.SATAG (for Fixed mode only) registers
6. Enable the AC97 mode of operation (AC97EN bit in SSI.SACNT register)

Once the SSI starts transmitting and receiving data (after being configured in AC97 mode), the programmer needs to service the interrupts, as and when they are raised (updates to command address/data or tag registers, reading of received data and writing more data for transmission). Further details regarding fixed and variable mode implementation are provided in the following sections.

While using AC97 in Two-Channel Mode (TCH\_EN=1), it is recommended that the received tag is not stored in the Rx FIFO (TIF=0). In case the programmer needs to update the SSI.SATAG register and also issue a RD/WR command (in a single frame), it is recommended that the SSI.SATAG register be updated prior to issuing a RD/WR command.

#### **48.8.1.5.1 AC97 Fixed Mode (SSI.SACNT[1]=0)**

In fixed mode of operation, SSI transmits in accordance with the AC97 Frame Rate Divider bits (i.e. FRDIV in SACNT) which decides the number of frames for which the SSI should be idle, after operating for one frame.

In a valid frame, TAG Value (written by Core) will be transmitted in Slot #0, Command Address will be transmitted in Slot #1 in case of RD/WR Command, and Command Data will be transmitted in Slot #2 in case of a WR Command. The data from TX-FIFO is transmitted in Slot #3 - Slot #12 depending on the valid slots indicated by the TAG value.

While receiving, bit 15 of the TAG Value (Slot #0) is checked to see if the CODEC is ready. If this bit is set, the frame is received. The received TAG provides the information about Slots containing valid data. The the corresponding TAG bit is valid, the Command Address (Slot #1) and Command Data (Slot #2) values are stored in the corresponding registers. The received data (Slot #3 - Slot #12) is then stored in the Rx-FIFO (for valid slots).

#### **48.8.1.5.2 AC97 Variable Mode (SSI.SACNT[1]=1)**

In Variable Mode, the transmit slots which should contain data in the current frame are determined by SLOTREQ bits received in the previous frame. While receiving, if the CODEC is ready, the frame is received and the SLOTREQ bits (contained in Slot #1) are stored for scheduling transmission in the next frame.

The SSI.SACCST, SSI.SACCEN and SSI.SACCDIS registers helps in determining which transmit slots are active. This information is used to ensure that SSI does not transmit data for powered-down/inactive channels.

## 48.8.2 External Frame and Clock Operation

When applying external frame sync and clock signals to SSI, there should be at least 4-bit clock cycles between the enabling of the transmit or receive section and the rising edge of the corresponding frame sync signal.

The transition of STFS or SRFS should be synchronized with the rising edge of external clock signal, STCK or SRCK.

### 48.8.2.1 Data Alignment Formats Supported

The SSI supports three data formats in order to provide flexibility with handling data. These formats dictate how data is written to (and read from) the data registers. Therefore, data can appear in different places in SSI.STX0/1 and SSI.SRX0/1 based on the data format and the number of bits per word. Independent data formats are supported for both the transmitter and receiver (that is, the transmitter and receiver can use different data formats).

The supported data formats are:

- MSB alignment
- LSB alignment
  - Zero-extended (receive data only)
  - Sign-extended (receive data only)

With MSB alignment, the most significant byte is bits 31 through 24 of the data register if the word length is larger than or equal to 16 bits. If the word length is less than 16 bits and MSB alignment is chosen, the most significant byte is bits 15 through 8. With LSB alignment, the least significant byte is bits 7 through 0. Data alignment is controlled by the TXBIT0 bit in the SSI.STCR and the RXBIT0 bit in the SSI.SRCR. See the table below for the bit assignment for all the data formats supported by the SSI.

**Table 48-6. Data Alignment**

Format	Bit Number																																
	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
8-bit LSB Aligned																										7	6	5	4	3	2	1	0
8-bit MSB Aligned																	7	6	5	4	3	2	1	0									
10-bit LSB Aligned																								9	8	7	6	5	4	3	2	1	0
10-bit MSB Aligned																	9	8	7	6	5	4	3	2	1	0							

*Table continues on the next page...*

**Table 48-6. Data Alignment (continued)**

12-bit LSB Aligned																		1	1	9	8	7	6	5	4	3	2	1	0										
12-bit MSB Aligned																		1	1	9	8	7	6	5	4	3	2	1	0										
16-bit LSB Aligned																		1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0						
16-bit MSB Aligned	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0																							
18-bit LSB Aligned																		1	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0					
18-bit MSB Aligned	1	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0																						
20-bit LSB Aligned																		1	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0					
20-bit MSB Aligned	1	1	1	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0																				
22-bit LSB Aligned																		2	2	1	1	1	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0	
22-bit MSB Aligned	2	2	1	1	1	1	1	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0																
24-bit LSB Aligned																		2	2	2	2	1	1	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0
24-bit MSB Aligned	2	2	2	2	1	1	1	1	1	1	1	1	1	1	9	8	7	6	5	4	3	2	1	0															

In addition, receive data can either be zero-extended or sign-extended if LSB alignment is selected. With zero-extension, all bits above the most significant bit are 0's. This format is useful when data is stored in a pure integer format. With sign-extension, all bits above the most significant bit are equal to the most significant bit. This format is useful when data is stored in a fixed-point integer format (which implies fractional values). Receive data extension is controlled by the RXEXT bit in the SSI.SRCR. Transmit data used with LSB alignment has no concept of sign/zero-extension. Unused bits above the most significant bit are simply ignored.

When configured in I2S or AC97 mode, the SSI forces the selection of LSB alignment. However, RXEXT still permits a choice between zero-extension and sign-extension.

See [SSI](#) for more details on the relevant bits in the SSI.STCR and SSI.SRCR registers.

### 48.8.3 SSI Architecture

The Synchronous Serial Interface (SSI) is connected to chip pads through the Digital Audio Mux (AUDMUX) block or directly as well. The AUDMUX can be configured to connect the SSI to the chip pads in various ways.

Refer to [Figure 48-1](#) for a block diagram of the SSI.

#### 48.8.4 SSI Clocking

The SSI uses the following clocks:

- Bit clock - Used to serially clock the data bits in and out of the SSI port. This clock is either generated internally (from SSI's sys clock) or taken from external clock source (through the Tx/Rx clock ports).
- Word clock - Used to count the number of data bits per word (8, 10, 12, 16, 18, 20, 22 or 24 bits). This clock is generated internally from the bit clock.
- Frame clock (Frame Sync) - Used to count the number of words in a frame. This signal can be generated internally from the bit clock, or taken from external source (from the Tx/Rx frame sync ports).
- Network clock - In master mode, this is an integer multiple of frame clock. This is oversampling clock. It is used in cases when SSI has to provide the clock.

Care should be taken to ensure that the bit clock frequency (either internally generated by dividing the SSI's sys clock or sourced from external device through Tx/Rx clock ports) is never greater than 1/5 of the `ipg_clk` (from CCM) frequency.

In Normal mode (`SCR[6:5]=00`), the bit clock, used to serially clock the data, is visible on the Serial Transmit Clock (STCK) and Serial Receive Clock (SRCK) ports. The word clock is an internal clock used to determine when transmission of an 8, 10, 12, 16, 18, 20, 22 or 24 bit word has completed. The word clock in turn then clocks the frame clock, which counts the number of words in the frame. The frame clock can be viewed on the STFS and SRFS frame sync ports, because a frame sync is generated after the correct number of words in the frame have passed. In master and synchronous mode, the unused port SRCK is used as network clock (oversampling clock) enabled by the SCR register bit 15, `SYS_CLK_EN`. This network clock is an oversampling clock of the frame sync clock (STFS). In this mode, the word length (WL), Prescaler Range (PSR), Prescaler Modulus (PM) and Frame rate (DC) selects the ratio of network clock to sampling clock STFS. In case of I2S mode, the network clock (oversampling clock) can be made available on this port if the `SYS_CLK_EN` bit is set. The relationship between the clocks and the dividers is shown in the figure below ("SSI Clocking"). The bit clock can be received from an SSI clock port or can be generated from the network clock through a divider, as shown in [Figure 48-22](#) ("SSI Transmit Clock Generator Block Diagram").

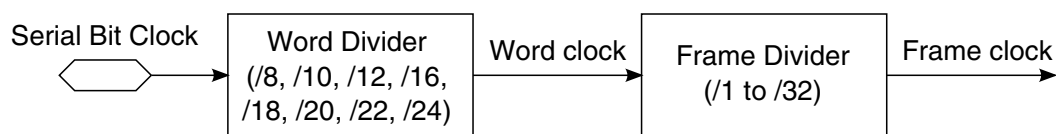


Figure 48-21. SSI Clocking

#### 48.8.4.1 SSI Clock and Frame Sync Generation

Data clock and frame sync signals can be generated internally, or can be obtained from external sources. If internally generated, the SSI clock generator is used to derive bit clock and frame sync signals from the SSI's sys clock. The SSI clock generator consists of a selectable, fixed prescaler and a programmable prescaler for bit rate clock generation.

In Gated Clock mode, the data clock is valid only when data is being transmitted. Otherwise the clock port is pulled to the inactive state. A programmable frame rate divider and a word length divider are used for frame rate sync signal generation.

The following figure shows a block diagram of the clock generator for the transmit section. The serial bit clock can be internal or external, depending on the Transmit Direction (TXDIR) bit in the SSI Transmit Configuration Register (SSI.STCR). The receive section contains an equivalent clock generator circuit.

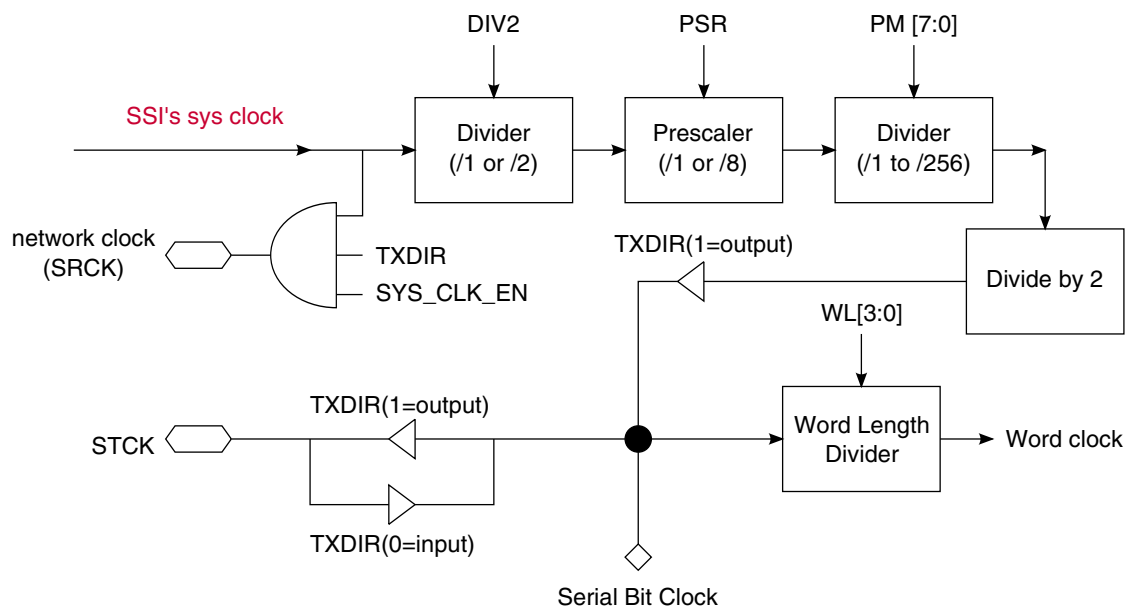


Figure 48-22. SSI Transmit Clock Generator Block Diagram

The figure below shows the Frame Sync Generator block for the transmit section. When internally generated, both receive and transmit frame sync are generated from the word clock and are defined by the Frame Rate Divider (DC[4:0]) bits and the Word Length (WL[3:0]) bits of the SSI Transmit Clock Control Register (SSI.STCCR). The receive section contains an equivalent circuit for the Frame Sync Generator.

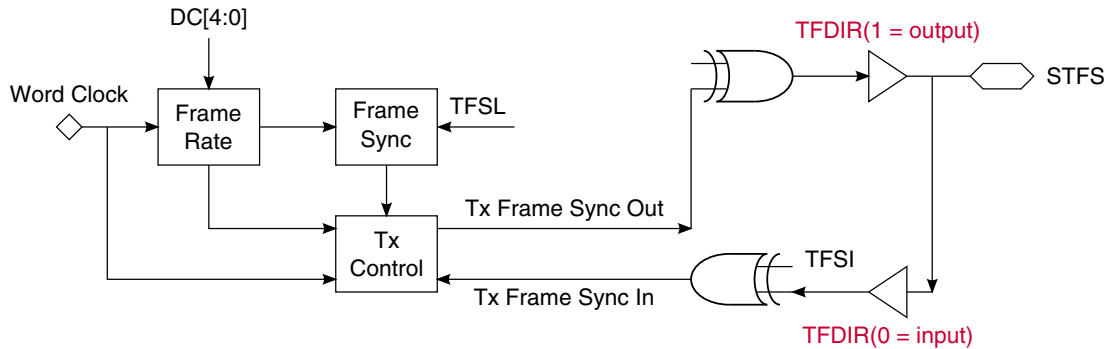


Figure 48-23. SSI Transmit Frame Sync Generator Block Diagram

#### 48.8.4.2 DIV2, PSR and PM Bit Description

The bit clock frequency can be calculated from the SSI's sys clock using the equation in the following figure.

#### NOTE

You must ensure that the bit-clock frequency must be never greater than 1/5 of the peripheral clock frequency. The oversampling clock frequency can go up to peripheral clock frequency. Bits DIV2, PSR and PM should not be all set to zero at the same time.

$$f_{\text{INT\_BIT\_CLK}} = f_{\text{SSI's sys clock}} / [(DIV2 + 1) \times (7 \times PSR + 1) \times (PM + 1) \times 2]$$

where PM = PM[7:0]

$$f_{\text{FRAME\_SYN\_CLK}} = (f_{\text{INT\_BIT\_CLK}}) / [(DC + 1) \times WL]$$

where DC = DC[4:0] and WL = 8, 10, 12, 16, 18, 20, 22, 24

Figure 48-24. SSI Bit Clock Equation



For example, if the SSI's sys clock is 12.288 MHz, in 8-bit word Normal mode with DC[4:0] set to 0(00000), PM[7:0] set to 47 (0010 1111), the PSR bit cleared, DIV2 bit set to 1, a bit clock rate of  $12.288 \text{ Mhz} / [1 \times 4 \times 48] = 64 \text{ kHz}$  is generated. Since the 8-bit word rate is equal to one (i.e. normal mode), the sampling rate (FS rate) would then be  $64 \text{ kHz} / [1 \times 8] = 8 \text{ kHz}$ .

In the next example, SSI's sys clock is 11.2896 Mhz. A 16-bit word Network mode with DC[4:0] set to 1 (00001), PM[7:0] set to 3 (0000 0011), the PSR bit is set to 0, DIV2 bit set to 0, and a 11.2896 MHz oversampling clock, a bit clock rate of  $11.2896 \text{ Mhz} / [1 \times 2 \times 4] = 1.4112 \text{ MHz}$  is generated. Since the 16-bit word rate is equal to two, the sampling rate (FS rate) would be  $1.4112 \text{ MHz} / [2 \times 16] = 44.1 \text{ kHz}$ .

The table below shows programming examples for the clock dividers in the CCM and the SSI to support various bit clock (STCK) frequencies.

**Table 48-7. SSI Bit Clock and Frame Rate as a Function of PSR, PM, and DIV2**

Bits/ Word	Words / Frame	Ideal Frame Rate (kHz)	PLL Freq (Mhz)	SSDIV (in CCM)	SSI's sys clock Freq (Mhz)	DIV2	PSR	PM	WL	DC	Actual Serial bit clock Freq (kHz) STCK	Target Serial bit clock Freq (kHz) STCK	Error (Hz)
16	1	8	688.128	56	12.288	0	0	47	7	0	128	128	0
16	2	8	688.128	56	12.288	0	0	23	7	1	256	256	0
16	4	8	688.128	56	12.288	0	0	11	7	3	512	512	0
16	1	12	688.128	56	12.288	0	0	31	7	0	192	192	0
16	2	12	688.128	56	12.288	0	0	15	7	1	384	384	0
16	4	12	688.128	56	12.288	0	0	7	7	3	768	768	0
16	1	16	688.128	56	12.288	0	0	23	7	0	256	256	0
16	2	16	688.128	56	12.288	0	0	11	7	1	512	512	0
16	4	16	688.128	56	12.288	0	0	5	7	3	1024	1024	0
16	1	24	688.128	56	12.288	0	0	15	7	0	384	384	0
16	2	24	688.128	56	12.288	0	0	7	7	1	768	768	0
16	4	24	688.128	56	12.288	0	0	3	7	3	1536	1536	0
16	1	32	688.128	56	12.288	0	0	11	7	0	512	512	0
16	2	32	688.128	56	12.288	0	0	5	7	1	1024	1024	0
16	4	32	688.128	56	12.288	0	0	2	7	3	2048	2048	0
16	1	48	688.128	56	12.288	0	0	15	7	0	768	768	0
16	2	48	688.128	56	12.288	0	0	3	7	1	1536	1536	0
16	4	48	688.128	56	12.288	0	0	1	7	3	3072	3072	0
16	1	11.025	632.217 6	56	11.2896	0	0	31	7	0	176.4	176.4	0

Table continues on the next page...

**Table 48-7. SSI Bit Clock and Frame Rate as a Function of PSR, PM, and DIV2 (continued)**

Bits/ Word	Words / Frame	Ideal Frame Rate (kHz)	PLL Freq (Mhz)	SSIDIV (in CCM)	SSI's sys clock Freq (Mhz)	DIV2	PSR	PM	WL	DC	Actual Serial bit clock Freq (kHz) STCK	Target Serial bit clock Freq (kHz) STCK	Error (Hz)
16	2	11.025	632.217 6	56	11.2896	0	0	15	7	1	352.8	352.8	0
16	4	11.025	632.217 6	56	11.2896	0	0	7	7	3	705.6	705.6	0
16	1	22.05	632.217 6	56	11.2896	0	0	15	7	0	352.8	352.8	0
16	2	22.05	632.217 6	56	11.2896	0	0	7	7	1	705.6	705.6	0
16	4	22.05	632.217 6	56	11.2896	0	0	3	7	3	1411.2	1411.2	0
16	1	44.1	632.217 6	56	11.2896	0	0	7	7	0	705.6	705.6	0
16	2	44.1	632.217 6	56	11.2896	0	0	3	7	1	1411.2	1411.2	0
16	4	44.1	632.217 6	56	11.2896	0	0	1	7	3	2822.4	2822.4	0

**NOTE**

The table above describes how various frame rates can be achieved with the PLLs supplying a frequency of 688.128 MHz and 633.2176 MHz (with WL and DC settings as shown).

These clocks are recommended as convenient starting points but the system allows for other input clock frequencies as well.

Table 48-7 shows programming of the CCM and SSI dividers in order to generate the appropriate network clock and serial bit clock frequencies for various sampling rates. In these examples, the master mode is selected either by setting I2S master bit (SCR[6:5]=01) or individually programming the SSI in network, synchronous, transmit internal mode (the table specifically illustrates the I2S mode frequencies/sample rates). The network clock is oversampling clock.

Note that the I2S master mode requires that a word length of 32 bits be used (regardless of the actual data type). Consequently, the fixed I2S frame rate of 64 bits per frame (word length (WL) can be any value) and DC of 1 are assumed.

### 48.8.5 Receive Interrupt Enable Bit Description

When the RIE and RE bit are set, the processor is interrupted when either of the SSI Receive FIFO Full (RFF0/1) bits in SSI.SISR is set (if the corresponding Receive FIFO is enabled).

If the Receive FIFO is not enabled, the interrupt is generated when the corresponding SSI Receive Data Ready (RDR0/1) bit in the SSI.SISR is set. When the receive FIFO is enabled, a maximum of 15 values are available to be read ( 15 values per channel in Two-Channel mode). If not enabled, then one value can be read from the SRX register (one each in case of Two-Channel mode). If the RIE bit is cleared, these interrupts are disabled. However, the RFF0/1 and RDR0/1 bits still indicate the receive data register full condition. Reading the SSI.SRX registers clears the RDR bits, thus clearing the pending interrupt. Two receive data interrupts (two per channel in case of Two-Channel mode) are available: receive data with exception status and receive data without exception. The tables below show the conditions under which these interrupts are generated.

**Table 48-8. SSI Receive Data 1 Interrupts**

Interrupt	RIE	ROE0	RFF0/RDR0
Receive Data 1(with Exception Status)	1	1	1
Receive Data 1(without exception)	1	0	1

**Table 48-9. SSI Receive Data 0 Interrupts**

Interrupt	RIE	ROE1	RFF1/RDR1
Receive Data 0 (with Exception Status)	1	1	1
Receive Data 0 (without exception)	1	0	1

### 48.8.6 Transmit Interrupt Enable Bit Description

The SSI Transmit Interrupt Enable (TIE) control bit determines whether the processor is interrupted when the SSI transmitter needs to be serviced.

When the TIE and TE bits are set, the program controller is interrupted when either of the SSI Transmit FIFO Empty (TFE0/1) flags in SISR are set (if corresponding Transmit FIFO is enabled). If the corresponding Transmit FIFO is not enabled, an interrupt is generated when the corresponding SSI Transmit Data Register Empty (TDE0/1) flag in the SISR is set and Transmit Enable (TE) bit is set.

When Transmit FIFO 0 is enabled, a maximum of 15 values can be written to the SSI ( 15 per channel in case of Two-Channel mode, using Tx FIFO 1). If not enabled, then one value can be written to the SSI.STX0 register (one per channel in case of Two-Channel mode using SSI.STX1). When the TIE bit is cleared, all transmit interrupts are disabled. However, the TDE0/1 bits always indicate the corresponding SSI.STX register empty condition, even when the transmitter is disabled by the Transmit Enable (TE) bit (in the SSI.SCR). Writing data to the STX clears the corresponding TDE bit, thus clearing the interrupt. Two transmit data interrupts are available (four in case of Two-Channel mode, two per channel): transmit data with exception status and transmit data without exceptions. The tables below show the conditions under which these interrupts are generated.

**Table 48-10. SSI Transmit Data 1 Interrupts**

Interrupt	TIE	TUE1	TFE1/TDE1
Transmit Data 1 (with Exception Status)	1	1	1
Transmit Data 1 (without exception)	1	0	1

**Table 48-11. SSI Transmit Data 0 Interrupts**

Interrupt	TIE	TUE0	TFE0/TDE0
Transmit Data 0 (with Exception Status)	1	1	1
Transmit Data 0 (without exception)	1	0	1

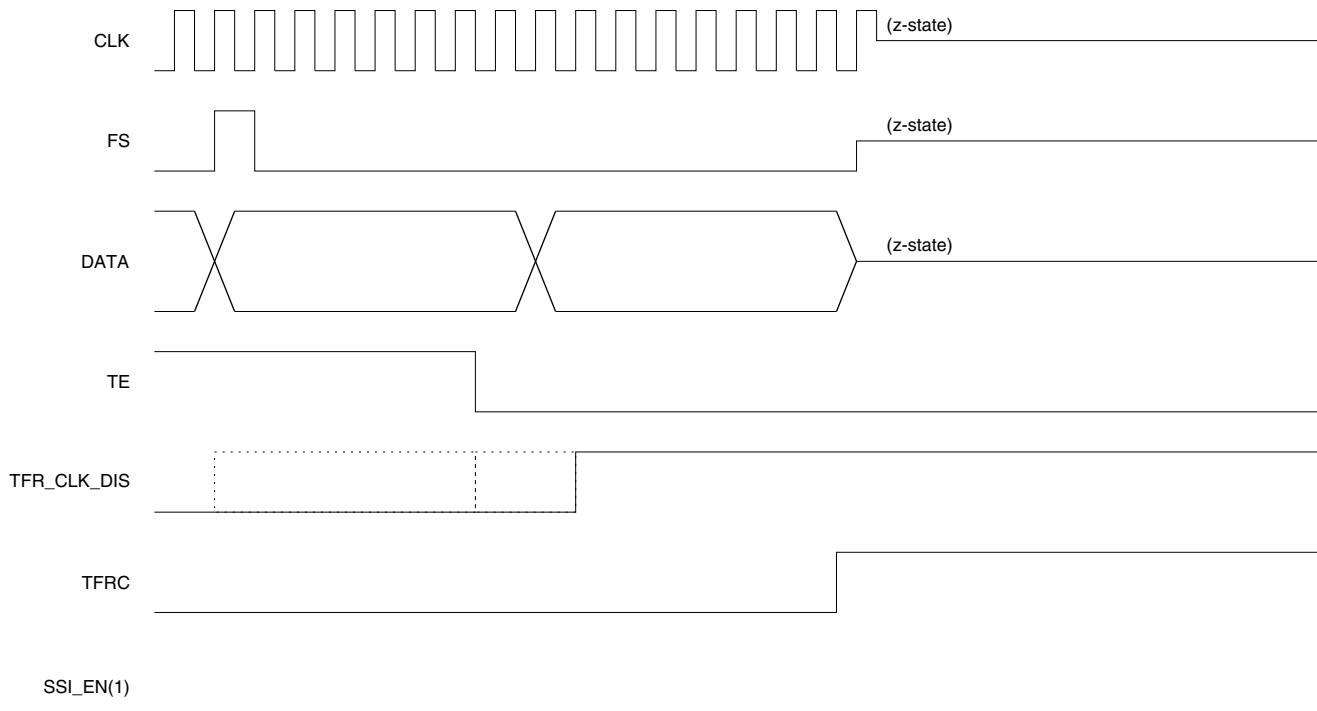
### 48.8.7 Internal Frame and Clock Shutdown

During transmit/receive operation, disabling TE/RE will ensure that data transmission/reception stops after current frame ends following which TFRC/RFRC Status bits will get set to indicate the Frame Completion State.

If TE is disabled 4 clock cycles before the next frame, extra frame generated are invalid frames. TFR\_CLK\_DIS/RFR\_CLK\_DIS bit is set in the current or any of the previous frames, SSI will stop driving the STFS/SRFS and STCK/SRCK signals after the current frame ends.

If TFR\_CLK\_DIS/RFR\_CLK\_DIS bit is not set, SSI will continue generating STFS/SRFS and STCK/SRCK signals (in case direction is from SSI), which then can be disabled by writing '1' to TFR\_CLK\_DIS/RFR\_CLK\_DIS bit. SSI will then stop driving these signals after end of frame is reached following which TFRC/RFRC status bits will get set to indicate the Frame Completion State.

The following figure is an illustration of transmission case where TXDIR and TFDIR are both set to '1'. In this case TE is disabled with TFR\_CLK\_DIS bit set in current or any of the previous frames.



**Figure 48-25. TFR\_CLK\_DIS assertion in current or previous frame as TE disable**

The figure below is an illustration of transmission case where TXDIR and TFDIR are both set to '1'. In this case TFR\_CLK\_DIS bit is set after few frames of disabling TE. TFRC (Transmit Frame Complete) is set at frame boundary after TE is cleared. Once software services this interrupt and sets TFR\_CLK\_DIS bit later, TFRC bit is again set at next frame boundary.

## Functional Description

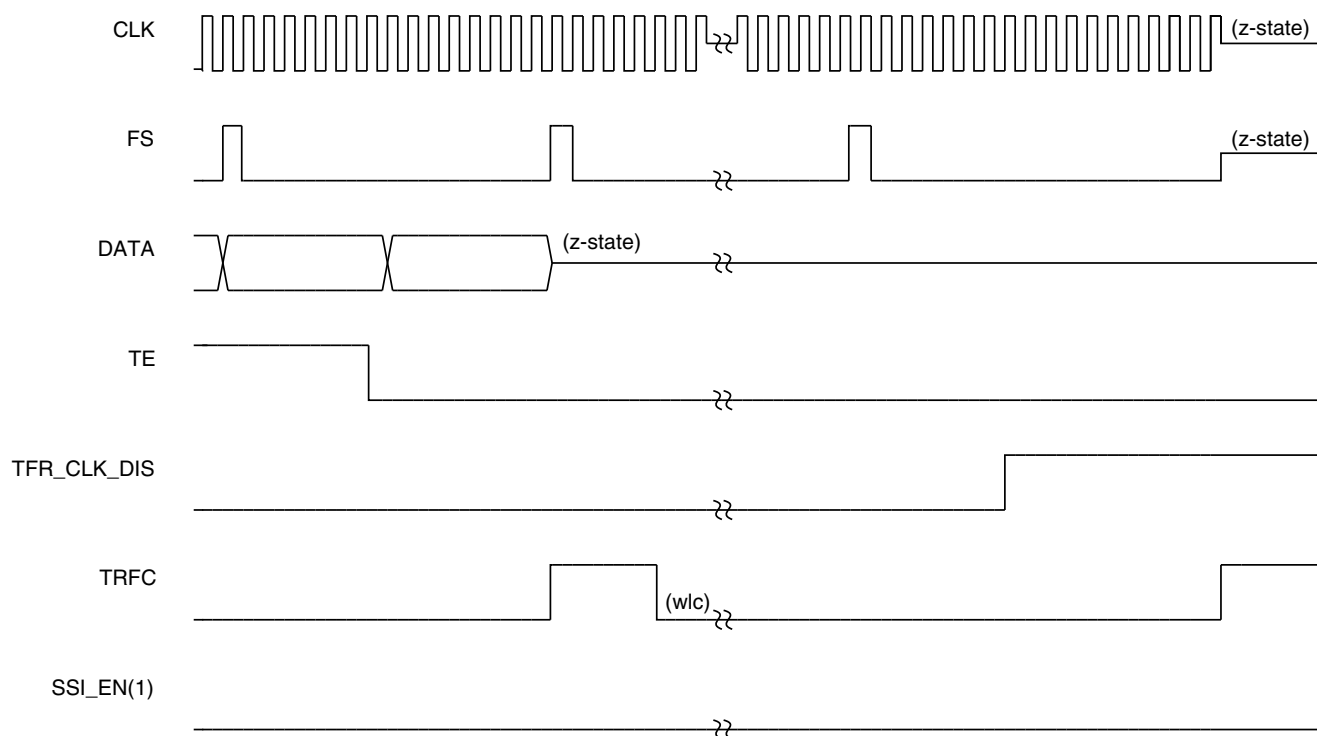


Figure 48-26. TFR\_CLK\_DIS assertion in subsequent frame after disabling TE

## 48.8.8 Peripheral Bus Interface

The SSI has a Peripheral Bus interface to provide a control and data interface. This interface is used by both the processor and DMA controller.

### 48.8.8.1 Transfer Lengths Supported

The Peripheral Bus interface of the SSI only supports 32-bit transfers with all SSI registers other than SSI.STX0, SSI.STX1, SSI.SRX0, and SSI.SRX1 (that is, the data registers).

With the exception of the data registers, using 8-bit and 16-bit transactions could result in undesired behavior but will not result in a transfer bus error. The data registers (SSI.STX0, SSI.STX1, SSI.SRX0, and SSI.SRX1) support 8-bit, 16-bit, and 32-bit transfer lengths without restrictions.

### 48.8.8.2 Transfer Bus Errors

Transfer bus errors are generated upon response to the following:

- Write transfer to a read-only register.
- Read or write access to a register space beyond the last populated register of the SSI in its memory map (up until the end of the allocated memory address range of the SSI).

### 48.8.8.3 Clock Rate

The Peripheral Bus clock frequency must be at least five times the serial bit clock frequency.

## 48.8.9 Reset

The SSI is affected by the following types of reset:

- Power-on Reset-The Power-on reset clears the SSIEN bit in SSI.SCR, which disables the SSI. All other status and control bits in the SSI are affected as described in SSI Programming Model in the "Memory Map and Register Definition section".
- SSI Reset-The SSI reset is generated when the SSIEN bit in the SSI.SCR is cleared. The SSI status bits are preset to the same state produced by the Power-on reset. The SSI control bits are unaffected. The control bits in the SSI.SCR are also unaffected. The SSI reset is useful for selective reset of the SSI without changing the present SSI control bits and without affecting the other peripherals.

## 48.9 SSI Memory Map/Register Definition

SSI memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
202_8000	SSI Transmit Data Register n (SSI1_STX0)	32	R/W	0000_0000h	<a href="#">48.9.1/2886</a>
202_8004	SSI Transmit Data Register n (SSI1_STX1)	32	R/W	0000_0000h	<a href="#">48.9.1/2886</a>
202_8008	SSI Receive Data Register n (SSI1_SRX0)	32	R	0000_0000h	<a href="#">48.9.2/2886</a>
202_800C	SSI Receive Data Register n (SSI1_SRX1)	32	R	0000_0000h	<a href="#">48.9.2/2886</a>
202_8010	SSI Control Register (SSI1_SCR)	32	R/W	0000_0000h	<a href="#">48.9.3/2887</a>
202_8014	SSI Interrupt Status Register (SSI1_SISR)	32	w1c	0000_3003h	<a href="#">48.9.4/2889</a>
202_8018	SSI Interrupt Enable Register (SSI1_SIER)	32	R/W	0000_3003h	<a href="#">48.9.5/2895</a>
202_801C	SSI Transmit Configuration Register (SSI1_STCR)	32	R/W	0000_0200h	<a href="#">48.9.6/2899</a>
202_8020	SSI Receive Configuration Register (SSI1_SRCR)	32	R/W	0000_0200h	<a href="#">48.9.7/2901</a>

Table continues on the next page...

## SSI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
202_8024	SSI Transmit Clock Control Register (SSI1_STCCR)	32	R/W	0004_0000h	<a href="#">48.9.8/2903</a>
202_8028	SSI Receive Clock Control Register (SSI1_SRCCR)	32	R/W	0004_0000h	<a href="#">48.9.9/2905</a>
202_802C	SSI FIFO Control/Status Register (SSI1_SFCSR)	32	R/W	0081_0081h	<a href="#">48.9.10/2906</a>
202_8038	SSI AC97 Control Register (SSI1_SACNT)	32	R/W	0000_0000h	<a href="#">48.9.11/2910</a>
202_803C	SSI AC97 Command Address Register (SSI1_SACADD)	32	R/W	0000_0000h	<a href="#">48.9.12/2911</a>
202_8040	SSI AC97 Command Data Register (SSI1_SACDAT)	32	R/W	0000_0000h	<a href="#">48.9.13/2911</a>
202_8044	SSI AC97 Tag Register (SSI1_SATAG)	32	R/W	0000_0000h	<a href="#">48.9.14/2912</a>
202_8048	SSI Transmit Time Slot Mask Register (SSI1_STMSK)	32	R/W	0000_0000h	<a href="#">48.9.15/2912</a>
202_804C	SSI Receive Time Slot Mask Register (SSI1_SRMSK)	32	R/W	0000_0000h	<a href="#">48.9.16/2913</a>
202_8050	SSI AC97 Channel Status Register (SSI1_SACCST)	32	R	0000_0000h	<a href="#">48.9.17/2913</a>
202_8054	SSI AC97 Channel Enable Register (SSI1_SACCEN)	32	W	0000_0000h	<a href="#">48.9.18/2914</a>
202_8058	SSI AC97 Channel Disable Register (SSI1_SACCDIS)	32	W	0000_0000h	<a href="#">48.9.19/2914</a>
202_C000	SSI Transmit Data Register n (SSI2_STX0)	32	R/W	0000_0000h	<a href="#">48.9.1/2886</a>
202_C004	SSI Transmit Data Register n (SSI2_STX1)	32	R/W	0000_0000h	<a href="#">48.9.1/2886</a>
202_C008	SSI Receive Data Register n (SSI2_SRX0)	32	R	0000_0000h	<a href="#">48.9.2/2886</a>
202_C00C	SSI Receive Data Register n (SSI2_SRX1)	32	R	0000_0000h	<a href="#">48.9.2/2886</a>
202_C010	SSI Control Register (SSI2_SCR)	32	R/W	0000_0000h	<a href="#">48.9.3/2887</a>
202_C014	SSI Interrupt Status Register (SSI2_SISR)	32	w1c	0000_3003h	<a href="#">48.9.4/2889</a>
202_C018	SSI Interrupt Enable Register (SSI2_SIER)	32	R/W	0000_3003h	<a href="#">48.9.5/2895</a>
202_C01C	SSI Transmit Configuration Register (SSI2_STCR)	32	R/W	0000_0200h	<a href="#">48.9.6/2899</a>
202_C020	SSI Receive Configuration Register (SSI2_SRCR)	32	R/W	0000_0200h	<a href="#">48.9.7/2901</a>
202_C024	SSI Transmit Clock Control Register (SSI2_STCCR)	32	R/W	0004_0000h	<a href="#">48.9.8/2903</a>
202_C028	SSI Receive Clock Control Register (SSI2_SRCCR)	32	R/W	0004_0000h	<a href="#">48.9.9/2905</a>
202_C02C	SSI FIFO Control/Status Register (SSI2_SFCSR)	32	R/W	0081_0081h	<a href="#">48.9.10/2906</a>
202_C038	SSI AC97 Control Register (SSI2_SACNT)	32	R/W	0000_0000h	<a href="#">48.9.11/2910</a>
202_C03C	SSI AC97 Command Address Register (SSI2_SACADD)	32	R/W	0000_0000h	<a href="#">48.9.12/2911</a>
202_C040	SSI AC97 Command Data Register (SSI2_SACDAT)	32	R/W	0000_0000h	<a href="#">48.9.13/2911</a>

Table continues on the next page...



## SSI memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
202_C044	SSI AC97 Tag Register (SSI2_SATAG)	32	R/W	0000_0000h	<a href="#">48.9.14/2912</a>
202_C048	SSI Transmit Time Slot Mask Register (SSI2_STMSK)	32	R/W	0000_0000h	<a href="#">48.9.15/2912</a>
202_C04C	SSI Receive Time Slot Mask Register (SSI2_SRMSK)	32	R/W	0000_0000h	<a href="#">48.9.16/2913</a>
202_C050	SSI AC97 Channel Status Register (SSI2_SACCST)	32	R	0000_0000h	<a href="#">48.9.17/2913</a>
202_C054	SSI AC97 Channel Enable Register (SSI2_SACCEN)	32	W	0000_0000h	<a href="#">48.9.18/2914</a>
202_C058	SSI AC97 Channel Disable Register (SSI2_SACCDIS)	32	W	0000_0000h	<a href="#">48.9.19/2914</a>
203_0000	SSI Transmit Data Register n (SSI3_STX0)	32	R/W	0000_0000h	<a href="#">48.9.1/2886</a>
203_0004	SSI Transmit Data Register n (SSI3_STX1)	32	R/W	0000_0000h	<a href="#">48.9.1/2886</a>
203_0008	SSI Receive Data Register n (SSI3_SRX0)	32	R	0000_0000h	<a href="#">48.9.2/2886</a>
203_000C	SSI Receive Data Register n (SSI3_SRX1)	32	R	0000_0000h	<a href="#">48.9.2/2886</a>
203_0010	SSI Control Register (SSI3_SCR)	32	R/W	0000_0000h	<a href="#">48.9.3/2887</a>
203_0014	SSI Interrupt Status Register (SSI3_SISR)	32	w1c	0000_3003h	<a href="#">48.9.4/2889</a>
203_0018	SSI Interrupt Enable Register (SSI3_SIER)	32	R/W	0000_3003h	<a href="#">48.9.5/2895</a>
203_001C	SSI Transmit Configuration Register (SSI3_STCR)	32	R/W	0000_0200h	<a href="#">48.9.6/2899</a>
203_0020	SSI Receive Configuration Register (SSI3_SRCR)	32	R/W	0000_0200h	<a href="#">48.9.7/2901</a>
203_0024	SSI Transmit Clock Control Register (SSI3_STCCR)	32	R/W	0004_0000h	<a href="#">48.9.8/2903</a>
203_0028	SSI Receive Clock Control Register (SSI3_SRCCR)	32	R/W	0004_0000h	<a href="#">48.9.9/2905</a>
203_002C	SSI FIFO Control/Status Register (SSI3_SFCSR)	32	R/W	0081_0081h	<a href="#">48.9.10/2906</a>
203_0038	SSI AC97 Control Register (SSI3_SACNT)	32	R/W	0000_0000h	<a href="#">48.9.11/2910</a>
203_003C	SSI AC97 Command Address Register (SSI3_SACADD)	32	R/W	0000_0000h	<a href="#">48.9.12/2911</a>
203_0040	SSI AC97 Command Data Register (SSI3_SACDAT)	32	R/W	0000_0000h	<a href="#">48.9.13/2911</a>
203_0044	SSI AC97 Tag Register (SSI3_SATAG)	32	R/W	0000_0000h	<a href="#">48.9.14/2912</a>
203_0048	SSI Transmit Time Slot Mask Register (SSI3_STMSK)	32	R/W	0000_0000h	<a href="#">48.9.15/2912</a>
203_004C	SSI Receive Time Slot Mask Register (SSI3_SRMSK)	32	R/W	0000_0000h	<a href="#">48.9.16/2913</a>
203_0050	SSI AC97 Channel Status Register (SSI3_SACCST)	32	R	0000_0000h	<a href="#">48.9.17/2913</a>
203_0054	SSI AC97 Channel Enable Register (SSI3_SACCEN)	32	W	0000_0000h	<a href="#">48.9.18/2914</a>
203_0058	SSI AC97 Channel Disable Register (SSI3_SACCDIS)	32	W	0000_0000h	<a href="#">48.9.19/2914</a>

## 48.9.1 SSI Transmit Data Register n (SSIx\_STXn)

### NOTE

Enable SSI (SSIEN=1) before writing to SSI Transmit Data Registers.

Address: Base address + 0h offset + (4d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STXn																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SSIx\_STXn field descriptions

Field	Description
31–0 STXn	<p>SSI Transmit Data. These bits store the data to be transmitted by the These are implemented as the first word of their respective Tx FIFOs. Data written to these registers is transferred to the Transmit Shift Register (TXSR), when shifting of the previous data is complete. If both FIFOs are in use, data is alternately transferred from STX0 and STX1, to TXSR. Multiple writes to the STX registers will not result in the previous data being over-written by the subsequent data. STX1 can only be used in Two-Channel mode of operation. Protection from over-writing is present irrespective of whether the transmitter is enabled or not.</p> <p>Example 1: If Tx FIFO0 is in use and user writes Data1...Data16 to STX0,Data16 will not over-write Data1. Data1...Data15 are stored in the FIFO whileData16 is discarded.</p> <p>Example 2: If Tx FIFO0 is not in use and user writes Data1, Data2 to STX0, then Data2 will not over-write Data1 and will be discarded.</p>

## 48.9.2 SSI Receive Data Register n (SSIx\_SRXn)

Address: Base address + 8h offset + (4d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SRXn																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SSIx\_SRXn field descriptions

Field	Description
31–0 SRXn	<p>SSI Receive Data. These bits store the data received by the These are implemented as the first word of their respective Rx FIFOs. These bits receive data from the RXSR depending on the mode of operation. In case both FIFOs are in use, data is transferred to each data register alternately. SRX1 can only be used in Two-Channel mode of operation.</p>

## 48.9.4 SSI Control Register (SSIx\_SCR)

The SSI Control Register (SSIx\_SCR) sets up the SSI reset is controlled by bit 0 in the SSI\_SCR. SSI operating modes are also selected in this register (except AC97 mode which is selected in the SSI\_SACNT register).

Address: Base address + 10h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			SYNC_TX_FS	RFR_CLK_DIS	TFR_CLK_DIS	CLK_IST	TCH_EN	SYS_CLK_EN	I2S_MODE		SYN	NET	RE	TE	SSIEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SSIx\_SCR field descriptions**

Field	Description
31–13 Reserved	This read-only field is reserved and always has the value 0.
12 SYNC_TX_FS	<p>SYNC_FS_TX bit provides a safe window for TE to be visible to the internal circuit which is just after FS occurrence. When SYNC_TX_FS is set, TE(SCR[1]) gets latched on FS occurrence &amp; latched TE is used to enable/disable SSI transmitter. TE needs setup of 2 bit-clock cycles before occurrence of FS. If TE is changed within 2 bit-clock cycles of FS occurrence, there is high probability that TE will be latched on next FS.</p> <p>Note: With TFR_CLK_DIS feature on, TE is used directly to enable transmitter in following cases (i) Sync mode &amp; Rx disabled (ii) Async Mode. Latched-TE is used to disable the transmitter.</p> <p>This bit has no relevance in gated mode and AC97 mode.</p> <p>0 <b>TE_NOT_LATCHED</b> — TE not latched with FS occurrence &amp; used directly for transmitter enable/disable.</p> <p>1 <b>TE_LATCHED</b> — TE latched with FS occurrence &amp; latched-TE used for transmitter enable/disable.</p>
11 RFR_CLK_DIS	<p>Receive Frame Clock Disable.</p> <p>This bit provides the option to keep the Frame-sync and Clock enabled or to disable them after the receive frame in which the receiver is disabled. Writing to this bit has effect only when RE is disabled. The receiver is disabled by clearing the RE bit.</p>

*Table continues on the next page...*

## SSIx\_SCR field descriptions (continued)

Field	Description
	<p>0 <b>CONTINUE</b> — Continue Frame-sync/Clock generation after current frame during which RE is cleared. This may be required when Frame-sync and Clocks are required from SSI, even when no data is to be received.</p> <p>1 <b>STOP</b> — Stop Frame-sync/Clock generation at next frame boundary. This will be effective also in case where receiver is already disabled in current or previous frames.</p>
10 TFR_CLK_DIS	<p>Transmit Frame Clock Disable.</p> <p>This bit provide option to keep the Frame-sync and Clock enabled or disabled after current transmit frame, in which transmitter is disabled by clearing TE bit. Writing to this bit has effect only when SSI is enabled TE is disabled.</p> <p>0 <b>CONTINUE</b> — Continue Frame-sync/Clock generation after current frame during which TE is cleared. This may be required when Frame-sync and Clocks are required from SSI, even when no data is to be received.</p> <p>1 <b>STOP</b> — Stop Frame-sync/Clock generation at next frame boundary. This will be effective also in case where transmitter is already disabled in current or previous frames.</p>
9 CLK_IST	<p>Clock Idle State. This bit controls the idle state of the transmit clock port during SSI internal gated mode.</p> <p>Note: When Clock idle state is '1' the clock polarity should always be negedge triggered and when Clock idle = '0' the clock polarity should always be positive edge triggered.</p> <p>0 <b>IDLE_0</b> — Clock idle state is '0'.</p> <p>1 <b>IDLE_1</b> — Clock idle state is '1'.</p>
8 TCH_EN	<p>Two-Channel Operation Enable. This bit allows SSI to operate in the two-channel mode. In this mode while receiving, the RXSR transfers data to SRX0 and SRX1 alternately and while transmitting, data is alternately transferred from STX0 and STX1 to TXSR. For an even number of slots, Two-Channel Operation can be enabled to optimize usage of both FIFOs or disabled as in the case of odd number of active slots. This feature is especially useful in I2S mode, where data for Left Speaker can be placed in Tx-FIFO0 and for Right speaker in Tx-FIFO1.</p> <p>0 <b>DISABLED</b> — Two-channel mode disabled.</p> <p>1 <b>ENABLED</b> — Two-channel mode enabled.</p>
7 SYS_CLK_EN	<p>Network Clock (Oversampling Clock) Enable. When set, this bit allows the SSI to output the network clock at the SRCK port, provided that synchronous mode, and transmit internal clock mode are set. The relationship between bit clock and network clock is determined by DIV2, PSR, and PM bits. This feature is especially useful in I2S Master mode to output network clock (oversampling clock) on SRCK port.</p> <p>0 <b>NOT_OUTPUT</b> — network clock not output on SRCK port.</p> <p>1 <b>OUTPUT</b> — network clock output on SRCK port.</p>
6–5 I2S_MODE	<p>I2S Mode Select. These bits allow the SSI to operate in Normal, I2S Master or I2S Slave mode. Refer to <a href="#">I2S Mode</a> for a detailed description of I2S Mode of operation. Refer to <a href="#">Table 48-5</a> for details regarding settings.</p>
4 SYN	<p>Synchronous Mode. This bit controls whether SSI is in synchronous mode or not. In synchronous mode, the transmit and receive sections of SSI share a common clock port (STCK) and frame sync port (STFS).</p> <p>0 <b>ASYNCH_MODE</b> — Asynchronous mode selected.</p> <p>1 <b>SYNCH_MODE</b> — Synchronous mode selected.</p>
3 NET	<p>Network Mode. This bit controls whether SSI is in network mode or not.</p> <p>0 <b>DISABLED</b> — Network mode not selected.</p> <p>1 <b>ENABLED</b> — Network mode selected.</p>

Table continues on the next page...

## SSIx\_SCR field descriptions (continued)

Field	Description
2 RE	<p>Receive Enable. This control bit enables the receive section of the SSI. When this bit is enabled, data reception starts with the arrival of the next frame sync. If data is being received when this bit is cleared, data reception continues until the end of the current frame and then stops. If this bit is set again before the second to last bit of the last time slot in the current frame, then reception continues without interruption. RE should not be toggled in the same frame.</p> <p>0 <b>DISABLED</b> — Receive section disabled. 1 <b>ENABLED</b> — Receive section enabled.</p>
1 TE	<p>Transmit Enable. This control bit enables the transmit section of the SSI. It enables the transfer of the contents of the STX registers to the TXSR and also enables the internal transmit clock. The transmit section is enabled when this bit is set and a frame boundary is detected. When this bit is cleared, the transmitter continues to send data until the end of the current frame and then stops. Data can be written to the STX registers with the TE bit cleared (the corresponding TDE bit will be cleared). If the TE bit is cleared and then set again before the second to last bit of the last time slot in the current frame, data transmission continues without interruption. The normal transmit enable sequence is to write data to the STX register(s) and then set the TE bit. The normal disable sequence is to clear the TE and TIE bits after the TDE bit is set.</p> <p>In gated clock mode, clearing the TE bit results in the clock stopping after the data currently in TXSR has shifted out. When the TE bit is set, the clock starts immediately (for internal gated clock mode). TE should not be toggled in the same frame.</p> <p>After enabling/disabling transmission, SSI expects 4 setup clock cycles before arrival of frame-sync for frame-sync to be accepted/rejected by In case of fewer clock cycles, there is high probability of the frame-sync to get missed.</p> <p>Note: If continuous clock is not provided, SSI expects 6 clock cycles before arrival of frame-sync for frame-sync to be accepted by</p> <p>0 <b>DISABLED</b> — Transmit section disabled. 1 <b>ENABLED</b> — Transmit section enabled.</p>
0 SSIEN	<p>SSIEN - SSI Enable</p> <p>This bit is used to enable/disable the SSI. When disabled, all SSI status bits are preset to the same state produced by the power-on reset, all control bits are unaffected, the contents of Tx and Rx FIFOs are cleared. When SSI is disabled, all internal clocks are disabled (except register access clock).</p> <p>0 <b>DISABLED</b> — SSI is disabled. 1 <b>ENABLED</b> — SSI is enabled.</p>

### 48.9.5 SSI Interrupt Status Register (SSIx\_SISR)

The SSI Interrupt Status Register (SSI\_SISR) is used to monitor the SSI. This register is used by the core to interrogate the status of the In gated mode of operation the TFS, RFS, TLS, RLS, TFRC and RFRC bits of AISR register are not generated. The status bits are described in the following table.

- SSI Status flags are valid when SSI is enabled.

## SSI Memory Map/Register Definition

- See [Receive Interrupt Enable Bit Description](#) and [Transmit Interrupt Enable Bit Description](#) for interrupt source mapping.
- All the flags in the SSI\_SISR are updated after the first bit of the next SSI word has completed transmission or reception. Certain status bits (ROE0/1 and TUE0/1) are cleared by writing 1 to the corresponding interrupt status bit in SSI\_SISR.

Address: Base address + 14h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							RFRC	TFRC	0				CMDAU	CMDDU	PXT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RDR1	RDR0	TDE1	TDE0	ROE1	ROE0	TUE1	TUE0	TFS	RFS	TLS	RLS	RFF1	RFF0	TFE1	TFE0
W					w1c	w1c	w1c	w1c								
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	1

### SSIx\_SISR field descriptions

Field	Description
31–25 Reserved	This read-only field is reserved and always has the value 0.
24 RFRC	Receive Frame Complete. This flag is set at the end of the frame during which Receiver is disabled. If Receive Frame & Clock are not disabled in the same frame, this flag is also set at the end of the frame in which Receive Frame & Clock are disabled. See the description of RFR_CLK_DIS bit for more details on how to disable Receiver Frame & Clock or keep them enabled after receiver is disabled.

Table continues on the next page...

**SSIx\_SISR field descriptions (continued)**

Field	Description
	<p>0 End of Frame not reached</p> <p>1 End of frame reached after disabling RE or disabling RFR_CLK_DIS, when receiver is already disabled.</p>
23 TFRC	<p>Transmit Frame Complete. This flag is set at the end of the frame during which Transmitter is disabled. If Transmit Frame &amp; Clock are not disabled in the same frame, this flag is also set at the end of the frame in which Transmit Frame &amp; Clock are disabled. See description of TFR_CLK_DIS bit for more details on how to disable Transmit Frame &amp; Clock or keep them enabled after transmitter is disabled.</p> <p>0 End of Frame not reached</p> <p>1 End of frame reached after disabling TE or disabling TFR_CLK_DIS, when transmitter is already disabled.</p>
22–19 Reserved	This read-only field is reserved and always has the value 0.
18 CMDAU	<p>Command Address Register Updated. This bit causes the Command Address Updated interrupt (when CMDAU_EN bit is set). This status bit is set each time there is a difference in the previous and current value of the received Command Address. This bit is cleared on reading the SACADD register.</p> <p>0 No change in SACADD register.</p> <p>1 SACADD register updated with different value.</p>
17 CMDDU	<p>Command Data Register Updated. This bit causes the Command Data Updated interrupt (when CMDDU_EN bit is set). This status bit is set each time there is a difference in the previous and current value of the received Command Data. This bit is cleared on reading the SACDAT register.</p> <p>0 No change in SACDAT register.</p> <p>1 SACDAT register updated with different value.</p>
16 RXT	<p>Receive Tag Updated. This status bit is set each time there is a difference in the previous and current value of the received tag. It causes the Receive Tag Interrupt (if RXT_EN bit is set). This bit is cleared on reading the SATAG register.</p> <p>0 No change in SATAG register.</p> <p>1 SATAG register updated with different value.</p>
15 RDR1	<p>Receive Data Ready 1. This flag bit is set when SRX1 or Rx FIFO 1 is loaded with a new value and Two-Channel mode is selected.</p> <p>RDR1 is cleared when the Core reads the SRX1 register. If Rx FIFO 1 is enabled, RDR1 is cleared when the FIFO is empty.</p> <p>If RIE and RDR1_EN are set, a Receive Data 1 interrupt request is issued on setting of RDR1 bit in case Rx FIFO1 is disabled, if the FIFO is enabled, the interrupt is issued on RFF1 assertion. The RDR1 bit is cleared by POR and SSI reset.</p> <p>0 No new data for Core to read.</p> <p>1 New data for Core to read.</p>
14 RDR0	<p>Receive Data Ready 0. This flag bit is set when SRX0 or Rx FIFO 0 is loaded with a new value.</p> <p>RDR0 is cleared when the Core reads the SRX0 register. If Rx FIFO 0 is enabled, RDR0 is cleared when the FIFO is empty.</p> <p>If RIE and RDR0_EN are set, a Receive Data 0 interrupt request is issued on setting of RDR0 bit in case Rx FIFO0 is disabled, if the FIFO is enabled, the interrupt is issued on RFF0 assertion. The RDR0 bit is cleared by POR and SSI reset.</p> <p>0 No new data for Core to read.</p> <p>1 New data for Core to read.</p>

*Table continues on the next page...*

**SSIx\_SISR field descriptions (continued)**

Field	Description
13 TDE1	<p>Transmit Data Register Empty 1. This flag is set whenever data is transferred to TXSR from STX1 register and Two-Channel mode is selected.</p> <p>If Tx FIFO1 is enabled, this occurs when there is at least one empty slot in STX1 or Tx FIFO1. If Tx FIFO1 is not enabled, this occurs when the contents of STX1 are transferred to TXSR.</p> <p>The TDE1 bit is cleared when the Core writes to STX1. If TIE and TDE1_EN are set, an SSI Transmit Data 1 interrupt request is issued on setting of TDE1 bit. The TDE1 bit is cleared by POR and SSI reset.</p> <p>0 Data available for transmission. 1 Data needs to be written by the Core for transmission.</p>
12 TDE0	<p>Transmit Data Register Empty 0. This flag is set whenever data is transferred to TXSR from STX0 register.</p> <p>If Tx FIFO 0 is enabled, this occurs when there is at least one empty slot in STX0 or Tx FIFO 0. If Tx FIFO 0 is not enabled, this occurs when the contents of STX0 are transferred to TXSR.</p> <p>The TDE0 bit is cleared when the Core writes to STX0. If TIE and TDE0_EN are set, an SSI Transmit Data 0 interrupt request is issued on setting of TDE0 bit. The TDE0 bit is cleared by POR and SSI reset.</p> <p>0 Data available for transmission. 1 Data needs to be written by the Core for transmission.</p>
11 ROE1	<p>Receiver Overrun Error 1. This flag is set when the RXSR is filled and ready to transfer to SRX1 register or to Rx FIFO 1 (when enabled) and these are already full and Two-Channel mode is selected. If Rx FIFO 1 is enabled, this is indicated by RFF1 flag, else this is indicated by the RDR1 flag. The RXSR is not transferred in this case.</p> <p>The ROE1 flag causes an interrupt if RIE and ROE1_EN are set.</p> <p>The ROE1 bit is cleared by POR and SSI reset. It is also cleared by writing '1' to this bit. Clearing the RE bit does not affect the ROE1 bit.</p> <p>0 Default interrupt issued to the Core. 1 Exception interrupt issued to the Core.</p>
10 ROE0	<p>Receiver Overrun Error 0. This flag is set when the RXSR is filled and ready to transfer to SRX0 register or to Rx FIFO 0 (when enabled) and these are already full. If Rx FIFO 0 is enabled, this is indicated by RFF0 flag, else this is indicated by the RDR0 flag. The RXSR is not transferred in this case.</p> <p>The ROE0 flag causes an interrupt if RIE and ROE0_EN are set.</p> <p>The ROE0 bit is cleared by POR and SSI reset. It is also cleared by writing '1' to this bit. Clearing the RE bit does not affect the ROE0 bit.</p> <p>0 Default interrupt issued to the Core. 1 Exception interrupt issued to the Core.</p>
9 TUE1	<p>Transmitter Underrun Error 1. This flag is set when the TXSR is empty (no data to be transmitted), the TDE1 flag is set, a transmit time slot occurs and the SSI is in Two-Channel mode. When a transmit underrun error occurs, the previous data is retransmitted. In Network mode, each time slot requires data transmission (unless masked through STMSK register), when the transmitter is enabled (TE is set).</p> <p>The TUE1 flag causes an interrupt if TIE and TUE1_EN are set.</p> <p>The TUE1 bit is cleared by POR and SSI reset. It is also cleared by writing '1' to this bit.</p> <p>0 Default interrupt issued to the Core. 1 Exception interrupt issued to the Core.</p>
8 TUE0	<p>Transmitter Underrun Error 0. This flag is set when the TXSR is empty (no data to be transmitted), the TDE0 flag is set and a transmit time slot occurs. When a transmit underrun error occurs, the previous data</p>

*Table continues on the next page...*



## SSIx\_SISR field descriptions (continued)

Field	Description
	<p>is retransmitted. In Network mode, each time slot requires data transmission (unless masked through STMSK register), when the transmitter is enabled (TE is set).</p> <p>The TUE0 flag causes an interrupt if TIE and TUE0_EN are set.</p> <p>The TUE0 bit is cleared by POR and SSI reset. It is also cleared by writing '1' to this bit.</p> <p>0 Default interrupt issued to the Core. 1 Exception interrupt issued to the Core.</p>
7 TFS	<p>Transmit Frame Sync. This flag indicates the occurrence of transmit frame sync. Data written to the STX registers during the time slot when the TFS flag is set, is sent during the second time slot (in Network mode) or in the next first time slot (in Normal mode). In Network mode, the TFS bit is set during transmission of the first time slot of the frame and is then cleared when starting transmission of the next time slot. In Normal mode, this bit is high for the first time slot. This flag causes an interrupt if TIE and TFS_EN are set. The TFS bit is cleared by POR and SSI reset.</p> <p>0 No Occurrence of Transmit frame sync. 1 Transmit frame sync occurred during transmission of last word written to STX registers.</p>
6 RFS	<p>Receive Frame Sync. This flag indicates the occurrence of receive frame sync. In Network mode, the RFS bit is set when the first slot of the frame is being received. It is cleared when the next slot begins to be received. In Normal mode, this bit is always high (When DC = 0). This flag causes an interrupt if RIE and RFS_EN are set. The RFS bit is cleared by POR and SSI reset.</p> <p>0 No Occurrence of Receive frame sync. 1 Receive frame sync occurred during reception of next word in SRX registers.</p>
5 TLS	<p>Transmit Last Time Slot. This flag indicates the last time slot in a frame. When set, it indicates that the current time slot is the last time slot of the frame. TLS is set at the start of the last transmit time slot and causes the SSI to issue an interrupt (if TIE and TLS_EN are set). TLS is not generated when frame rate is 1 in normal mode of operation. TLS is cleared when the SISR is read with this bit set. The TLS bit is cleared by POR and SSI reset.</p> <p>0 Current time slot is not last time slot of frame. 1 Current time slot is the last transmit time slot of frame.</p>
4 RLS	<p>Receive Last Time Slot. This flag indicates the last time slot in a frame. When set, it indicates that the current time slot is the last receive time slot of the frame. RLS is set at the end of the last time slot and causes the SSI to issue an interrupt (if RIE and RLS_EN are set). RLS is cleared when the SISR is read with this bit set. The RLS bit is cleared by POR and SSI reset.</p> <p>0 Current time slot is not last time slot of frame. 1 Current time slot is the last receive time slot of frame.</p>
3 RFF1	<p>Receive FIFO Full 1. This flag is set when Rx FIFO1 is enabled, the data level in Rx FIFO1 reaches the selected Rx FIFO WaterMark 1 (RFWM1) threshold and the SSI is in Two-Channel mode. The setting of RFF1 only causes an interrupt when RIE and RFF1_EN are set, Rx FIFO1 is enabled and the Two-Channel mode is selected. RFF1 is automatically cleared when the amount of data in Rx FIFO1 falls below the threshold. The RFF1 bit is cleared by POR and SSI reset.</p> <p>When Rx FIFO1 contains 15 words, the maximum it can hold, all further data received (for storage in this FIFO) is ignored until the FIFO contents are read.</p> <p>0 <b>NOT_FULL</b> — Space available in Receive FIFO1. 1 <b>FULL</b> — Receive FIFO1 is full.</p>
2 RFF0	<p>Receive FIFO Full 0. This flag is set when Rx FIFO0 is enabled and the data level in Rx FIFO0 reaches the selected Rx FIFO WaterMark 0 (RFWM0) threshold. The setting of RFF0 only causes an interrupt</p>

Table continues on the next page...

**SSIx\_SISR field descriptions (continued)**

Field	Description
	<p>when RIE and RFF0_EN are set and Rx FIFO0 is enabled. RFF0 is automatically cleared when the amount of data in Rx FIFO0 falls below the threshold. The RFF0 bit is cleared by POR and SSI reset.</p> <p>When Rx FIFO0 contains 15 words, the maximum it can hold, all further data received (for storage in this FIFO) is ignored until the FIFO contents are read.</p> <p>0 <b>NOT_FULL</b> — Space available in Receive FIFO0.  1 <b>FULL</b> — Receive FIFO0 is full.</p>
1 TFE1	<p>Transmit FIFO Empty 1. This flag is set when the empty slots in Tx FIFO exceed or are equal to the selected Tx FIFO WaterMark 1 (TFWM1) threshold and the Two-Channel mode is selected. The setting of TFE1 only causes an interrupt when TIE and TFE1_EN are set, Tx FIFO1 is enabled and Two-Channel mode is selected. The TFE1 bit is automatically cleared when the data level in Tx FIFO1 becomes more than the amount specified by the watermark bits. The TFE1 bit is set by POR and SSI reset.</p> <p>0 <b>HAS_DATA</b> — Transmit FIFO1 has data for transmission.  1 <b>EMPTY</b> — Transmit FIFO1 is empty.</p>
0 TFE0	<p>Transmit FIFO Empty 0. This flag is set when the empty slots in Tx FIFO exceed or are equal to the selected Tx FIFO WaterMark 0 (TFWM0) threshold. The setting of TFE0 only causes an interrupt when TIE and TFE0_EN are set and Tx FIFO0 is enabled. The TFE0 bit is automatically cleared when the data level in Tx FIFO0 becomes more than the amount specified by the watermark bits. The TFE0 bit is set by POR and SSI reset.</p> <p>0 <b>HAS_DATA</b> — Transmit FIFO0 has data for transmission.  1 <b>EMPTY</b> — Transmit FIFO0 is empty.</p>

## 48.9.6 SSI Interrupt Enable Register (SSIx\_SIER)

The SSI Interrupt Enable Register (SIER) is a 25-bit register used to set up the SSI interrupts and DMA requests.

Address: Base address + 18h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0							RFRCE	TFRCE	RDMAE	RIE	TDMAE	TIE	CMDAUIE	CMDDUIE	RXTIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RDR1IE	RDR0IE	TDE1IE	TDE0IE	ROE1IE	ROE0IE	TUE1IE	TUE0IE	TFSIE	RFSIE	TLSIE	RLSIE	RFF1IE	RFF0IE	TFE1IE	TFE0IE
W																
Reset	0	0	1	1	0	0	0	0	0	0	0	0	0	0	1	1

**SSIx\_SIER field descriptions**

Field	Description
31–25 Reserved	This read-only field is reserved and always has the value 0.
24 RFRCE	Receive Frame Complete Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not. 0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
23 TFRCE	Transmit Frame Complete Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not.

*Table continues on the next page...*

**SSIx\_SIER field descriptions (continued)**

Field	Description
	0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
22 RDMAE	Receive DMA Enable. This bit allows SSI to request for DMA transfers. When enabled, DMA requests are generated when any of the RFF0/1 bits in the SISR are set and if the corresponding RFEN bit is also set. If the corresponding FIFO is disabled, a DMA request is generated when the corresponding RDR bit is set.  0 SSI Receiver DMA requests disabled. 1 SSI Receiver DMA requests enabled.
21 RIE	Receive Interrupt Enable. This control bit allows the SSI to issue receiver related interrupts to the Core. Refer to <a href="#">Receive Interrupt Enable Bit Description</a> for a detailed description of this bit.  0 SSI Receiver Interrupt requests disabled. 1 SSI Receiver Interrupt requests enabled.
20 TDMAE	Transmit DMA Enable. This bit allows SSI to request for DMA transfers. When enabled, DMA requests are generated when any of the TFE0/1 bits in the SISR are set and if the corresponding TFEN bit is also set. If the corresponding FIFO is disabled, a DMA request is generated when the corresponding TDE bit is set.  0 SSI Transmitter DMA requests disabled. 1 SSI Transmitter DMA requests enabled.
19 TIE	Transmit Interrupt Enable. This control bit allows the SSI to issue transmitter data related interrupts to the Core. Refer to <a href="#">Transmit Interrupt Enable Bit Description</a> for a detailed description of this bit.  0 SSI Transmitter Interrupt requests disabled. 1 SSI Transmitter Interrupt requests enabled.
18 CMDAUIE	Command Address Register Updated Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not.  0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
17 CMDDUIE	Command Data Register Updated Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not.  0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
16 RXTIE	Receive Tag Updated Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not.  0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
15 RDR1IE	Receive Data Ready 1 Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not.  0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
14 RDR0IE	Receive Data Ready 0 Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not.

*Table continues on the next page...*

**SSIx\_SIER field descriptions (continued)**

Field	Description
	0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
13 TDE1IE	Transmit Data Register Empty 1 Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not.  0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
12 TDE0IE	Transmit Data Register Empty 0 Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not.  0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
11 ROE1IE	Receiver Overrun Error 1 Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not.  0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
10 ROE0IE	Receiver Overrun Error 0 Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not.  0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
9 TUE1IE	Transmitter Underrun Error 1 Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not.  0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
8 TUE0IE	Transmitter Underrun Error 0 Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not.  0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
7 TFSIE	Transmit Frame Sync Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not.  0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
6 RFSIE	Receive Frame Sync Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not.  0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
5 TLSIE	Transmit Last Time Slot Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not.

*Table continues on the next page...*

**SSIx\_SIER field descriptions (continued)**

Field	Description
	0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
4 RLSIE	Receive Last Time Slot Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not. 0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
3 RFF1IE	Receive FIFO Full 1 Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not. 0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
2 RFF0IE	Receive FIFO Full 0 Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not. 0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
1 TFE1IE	Transmit FIFO Empty 1 Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not. 0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.
0 TFE0IE	Transmit FIFO Empty 0 Interrupt Enable. Enable Bit. Controls whether the corresponding status bit in SISR can issue an interrupt to the core or not. 0 Corresponding status bit cannot issue interrupt. 1 Corresponding status bit can issue interrupt.

## 48.9.7 SSI Transmit Configuration Register (SSIx\_STCR)

The SSI Transmit Configuration Register (SSIx\_STCR) is a read/write control register used to direct the transmit operation of the STCR controls the direction of the bit clock and frame sync ports, STCK and STFS. Interrupt enable bit for the transmit sections is provided in this control register. The Power-on reset clears all SSI\_STCR bits. However, SSI reset does not affect the SSI\_STCR bits. The SSI\_STCR bits are described in the following paragraphs. See the Programmable Registers section for the programming model of the SSI. The SSI Control Register (SSI\_SCR) must first be set to enable interrupts. Next, the SSI interrupt bit in the Interrupt Enable Register (SSI\_SIER) must be set to enable the interrupt. Finally, the interrupt can be enabled from within the

Address: Base address + 1Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0							TXBIT0	TFEN1	TFEN0	TFDIR	TXDIR	TSHFD	TCKP	TFSL	TEFS
W																
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

**SSIx\_STCR field descriptions**

Field	Description
31–10 Reserved	This read-only field is reserved and always has the value 0.
9 TXBIT0	Transmit Bit 0. This control bit allows SSI to transmit the data word from bit position 0 or 15/31 in the transmit shift register. The shifting data direction can be MSB or LSB first, controlled by the TSHFD bit.  0 <b>MSB_ALIGNED</b> — Shifting with respect to bit 31 (if word length = 16, 18, 20, 22 or 24) or bit 15 (if word length = 8, 10 or 12) of transmit shift register (MSB aligned). 1 <b>LSB_ALIGNED</b> — Shifting with respect to bit 0 of transmit shift register (LSB aligned).
8 TFEN1	Transmit FIFO Enable 1. This bit enables transmit FIFO 1. When enabled, the FIFO allows 15 samples to be transmitted by the SSI (per channel) (a 9th sample can be shifting out) before TDE1 bit is set. When the FIFO is disabled, an interrupt is generated when a single sample is transferred to the transmit shift register (provided the interrupt is enabled).  0 Transmit FIFO 1 disabled. 1 Transmit FIFO 1 enabled.

*Table continues on the next page...*

## SSIx\_STCR field descriptions (continued)

Field	Description
7 TFENO	Transmit FIFO Enable 0. This bit enables transmit FIFO 0. When enabled, the FIFO allows 15 samples to be transmitted by the SSI per channel (a 9th sample can be shifting out) before TDE0 bit is set. When the FIFO is disabled, an interrupt is generated when a single sample is transferred to the transmit shift register (provided the interrupt is enabled).  0 Transmit FIFO 0 disabled. 1 Transmit FIFO 0 enabled.
6 TFDIR	Transmit Frame Direction. This bit controls the direction and source of the transmit frame sync signal. Internally generated frame sync signal is sent out through the STFS port and external frame sync is taken from the same port.  0 <b>EXTERNAL</b> — Frame Sync is external. 1 <b>INTERNAL</b> — Frame Sync generated internally.
5 TXDIR	Transmit Clock Direction. This bit controls the direction and source of the clock signal used to clock the TXSR. Internally generated clock is output through the STCK port. External clock is taken from this port. Refer to <a href="#">Table 48-2</a> for details of clock pin configurations.  0 <b>EXTERNAL</b> — Transmit Clock is external. 1 <b>INTERNAL</b> — Transmit Clock generated internally.
4 TSHFD	Transmit Shift Direction. This bit controls whether the MSB or LSB will be transmitted first in a sample.  <b>NOTE:</b> The CODEC device labels the MSB as bit 0, whereas the Core labels the LSB as bit 0. Therefore, when using a standard CODEC, Core MSB (CODEC LSB) is shifted in first (TSHFD cleared).  0 <b>MSB_FIRST</b> — Data transmitted MSB first. 1 <b>LSB_FIRST</b> — Data transmitted LSB first.
3 TSCKP	Transmit Clock Polarity. This bit controls which bit clock edge is used to clock out data for the transmit section. Note: TSCKP is 0 CLK_IST = 0; TSCKP is 1 CLK_IST = 1  0 <b>RISING_EDGE</b> — Data clocked out on rising edge of bit clock. 1 <b>FALLING_EDGE</b> — Data clocked out on falling edge of bit clock.
2 TFSI	Transmit Frame Sync Invert. This bit controls the active state of the frame sync I/O signal for the transmit section of SSI.  0 <b>ACTIVE_HIGH</b> — Transmit frame sync is active high. 1 <b>ACTIVE_LOW</b> — Transmit frame sync is active low.
1 TFSL	Transmit Frame Sync Length. This bit controls the length of the frame sync signal to be generated or recognized for the transmit section. The length of a word-long frame sync is same as the length of the data word selected by WL[3:0].  0 <b>ONE_WORD</b> — Transmit frame sync is one-word long. 1 <b>ONE_CLOCK_BIT</b> — Transmit frame sync is one-clock-bit long.
0 TEFS	Transmit Early Frame Sync. This bit controls when the frame sync is initiated for the transmit section. The frame sync signal is deasserted after one bit-for-bit length frame sync and after one word-for-word length frame sync. In case of synchronous operation, the frame sync can also be initiated on receiving the first bit of data.  0 <b>FIRST_BIT</b> — Transmit frame sync initiated as the first bit of data is transmitted. 1 <b>ONE_BIT_BEFORE</b> — Transmit frame sync is initiated one bit before the data is transmitted.



### 48.9.8 SSI Receive Configuration Register (SSIx\_SRCR)

The SSI Receive Configuration Register (SSI\_SRCR) is a read/write control register used to direct the receive operation of the SSI. SSI\_SRCR controls the direction of the bit clock and frame sync ports, SRCK and SRFS. Interrupt enable bit for the transmit sections is provided in this control register. The Power-on reset clears all SSI\_SRCR bits. However, SSI reset does not affect the SSI\_SRCR bits.

Address: Base address + 20h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0															
W						RXEXT	RXBIT0	RFEN1	RFEN0	RFDIR	RXDIR	RSHFD	RSCKP	RFSI	RFSL	REFS
Reset	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

#### SSIx\_SRCR field descriptions

Field	Description
31–11 Reserved	This read-only field is reserved and always has the value 0.
10 RXEXT	Receive Data Extension. This control bit allows SSI to store the received data word in sign extended form. This bit affects data storage only in case received data is LSB aligned (SRCR[9]=1)  0 <b>OFF</b> — Sign extension turned off. 1 <b>ON</b> — Sign extension turned on.
9 RXBIT0	Receive Bit 0. This control bit allows SSI to receive the data word at bit position 0 or 15/31 in the receive shift register. The shifting data direction can be MSB or LSB first, controlled by the RSHFD bit.  0 <b>MSB_ALIGNED</b> — Shifting with respect to bit 31 (if word length = 16, 18, 20, 22 or 24) or bit 15 (if word length = 8, 10 or 12) of receive shift register (MSB aligned). 1 <b>LSB_ALIGNED</b> — Shifting with respect to bit 0 of receive shift register (LSB aligned).
8 RFEN1	Receive FIFO Enable 1. This bit enables receive FIFO 1. When enabled, the FIFO allows 15 samples to be received by the SSI per channel (a 16th sample can be shifting in) before RDR1 bit is set. When the FIFO is disabled, an interrupt is generated when a single sample is received by the SSI (provided the interrupt is enabled).  0 Receive FIFO 1 disabled. 1 Receive FIFO 1 enabled.

Table continues on the next page...

## SSIx\_SRCR field descriptions (continued)

Field	Description
7 RFEN0	Receive FIFO Enable 0. This bit enables receive FIFO 0. When enabled, the FIFO allows 15 samples to be received by the SSI (per channel) (a 16th sample can be shifting in) before RDR0 bit is set. When the FIFO is disabled, an interrupt is generated when a single sample is received by the SSI (provided the interrupt is enabled).  0 Receive FIFO 0 disabled. 1 Receive FIFO 0 enabled.
6 RFDIR	Receive Frame Direction. This bit controls the direction and source of the receive frame sync signal. Internally generated frame sync signal is sent out through the SRFS port and external frame sync is taken from the same port.  0 <b>EXTERNAL</b> — Frame Sync is external. 1 <b>INTERNAL</b> — Frame Sync generated internally.
5 RXDIR	Receive Clock Direction. This bit controls the direction and source of the clock signal used to clock the RXSR. Internally generated clock is output through the SRCK port. External clock is taken from this port. Refer to <a href="#">Table 48-2</a> for details on clock pin configurations.  0 <b>EXTERNAL</b> — Receive Clock is external. 1 <b>INTERNAL</b> — Receive Clock generated internally.
4 RSHFD	Receive Shift Direction. This bit controls whether the MSB or LSB will be received first in a sample.  <b>NOTE:</b> The CODEC device labels the MSB as bit 0, whereas the Core labels the LSB as bit 0. Therefore, when using a standard CODEC, Core MSB (CODEC LSB) is shifted in first (RSHFD cleared).  0 <b>MSB_FIRST</b> — Data received MSB first. 1 <b>LSB_FIRST</b> — Data received LSB first.
3 RSCKP	Receive Clock Polarity. This bit controls which bit clock edge is used to latch in data for the receive section.  0 <b>FALLING_EDGE</b> — Data latched on falling edge of bit clock. 1 <b>RISING_EDGE</b> — Data latched on rising edge of bit clock.
2 RFSI	Receive Frame Sync Invert. This bit controls the active state of the frame sync I/O signal for the receive section of SSI.  0 <b>ACTIVE_HIGH</b> — Receive frame sync is active high. 1 <b>ACTIVE_LOW</b> — Receive frame sync is active low.
1 RFSL	Receive Frame Sync Length. This bit controls the length of the frame sync signal to be generated or recognized for the receive section. The length of a word-long frame sync is same as the length of the data word selected by WL[3:0].  0 <b>ONE_WORD</b> — Receive frame sync is one-word long. 1 <b>ONE_CLOCK_BIT</b> — Receive frame sync is one-clock-bit long.
0 REFS	Receive Early Frame Sync. This bit controls when the frame sync is initiated for the receive section. The frame sync is disabled after one bit-for-bit length frame sync and after one word-for-word length frame sync.  0 <b>FIRST_BIT</b> — Receive frame sync initiated as the first bit of data is received. 1 <b>ONE_BIT_BEFORE</b> — Receive frame sync is initiated one bit before the data is received.

### 48.9.9 SSI Transmit Clock Control Register (SSIx\_STCCR)

The SSI Transmit and Receive Control (SSI\_STCCR and SSI\_SRCCR) registers are 19-bit, read/write control registers used to direct the operation of the SSI. The Clock Controller Module (CCM) can source the SSI clock (SSI's sys clock from CCM's ssi\_clk\_root) from multiple sources and perform fractional division to support commonly used audio bit rates. The CCM can maintain the SSI's sys clock frequency at a constant rate even in cases where the ipg\_clk (from CCM) frequency changes. These registers control the SSI clock generator, bit and frame sync rates, word length, and number of words per frame for the serial data. The SSI\_STCCR register is dedicated to the transmit section, and the SSI\_SRCCR register is dedicated to the receive section except in Synchronous mode, in which the SSI\_STCCR register controls both the receive and transmit sections. Power-on reset clears all SSI\_STCCR and SSI\_SRCCR bits. SSI reset does not affect the SSI\_STCCR and SSI\_SRCCR bits. The control bits are described in the following paragraphs. Although the bit patterns of the SSI\_STCCR and SSI\_SRCCR registers are the same, the contents of these two registers can be programmed differently.

**Table 48-52. SSI Data Length**

WL3	WL2	WL1	WL0	Number of Bits/Word	Supported in Implementation
0	0	0	0	2	No
0	0	0	1	4	No
0	0	1	0	6	No
0	0	1	1	8	Yes
0	1	0	0	10	Yes
0	1	0	1	12	Yes
0	1	1	0	14	No
0	1	1	1	16	Yes
1	0	0	0	18	Yes
1	0	0	1	20	Yes
1	0	1	0	22	Yes
1	0	1	1	24	Yes
1	1	0	0	26	No
1	1	0	1	28	No
1	1	1	0	30	No
1	1	1	1	32	No

Address: Base address + 24h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													DIV2	PSR	WL3_WL0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

## SSI Memory Map/Register Definition

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SSIx\_STCCR field descriptions

Field	Description
31–19 Reserved	This read-only field is reserved and always has the value 0.
18 DIV2	Divide By 2. This bit controls a divide-by-two divider in series with the rest of the prescalers. 0 Divider bypassed. 1 Divider used to divide clock by 2.
17 PSR	Prescaler Range. This bit controls a fixed divide-by-eight prescaler in series with the variable prescaler. It extends the range of the prescaler for those cases where a slower bit clock is required. 0 Prescaler bypassed. 1 Prescaler used to divide clock by 8.
16–13 WL3_WL0	Word Length Control. These bits are used to control the length of the data words being transferred by the SSI. These bits control the Word Length Divider in the Clock Generator. They also control the frame sync pulse length when the FSL bit is cleared. In I2S Master mode, the SSI works with a fixed word length of 32, and the WL bits are used to control the amount of valid data in those 32 bits. In AC97 Mode of operation, if word length is set to any value other than 16 bits, it will result in a word length of 20 bits.
12–8 DC4_DC0	Frame Rate Divider Control. These bits are used to control the divide ratio for the programmable frame rate dividers. The divide ratio works on the word clock. In Normal mode, this ratio determines the word transfer rate. In Network mode, this ratio sets the number of words per frame. The divide ratio ranges from 1 to 32 in Normal mode and from 2 to 32 in Network mode.  In Normal mode, a divide ratio of 1 (DC=00000) provides continuous periodic data word transfer. A bit-length frame sync must be used in this case.  These bits can be programmed with values ranging from "00000" to "11111" to control the number of words in a frame.
7–0 PM7_PM0	Prescaler Modulus Select. These bits control the prescale divider in the clock generator. This prescaler is used only in Internal Clock mode to divide the internal clock. The bit clock output is available at the clock port.  A divide ratio from 1 to 256 (PM[7:0] = 0x00 to 0xFF) can be selected. Refer to <a href="#">DIV2, PSR and PM Bit Description</a> for details regarding settings.

### 48.9.10 SSI Receive Clock Control Register (SSIx\_SRCCR)

The SSI Transmit and Receive Control (SSI\_STCCR and SSI\_SRCCR) registers are 19-bit, read/write control registers used to direct the operation of the SSI. The Clock Controller Module (CCM) can source the SSI clock (SSI's sys clock-from CCM's ssi\_clk\_root) from multiple sources and perform fractional division to support commonly used audio bit rates. The CCM can maintain the SSI's sys clock frequency at a constant rate even in cases where the ipg\_clk from CCM frequency changes. These registers control the SSI clock generator, bit and frame sync rates, word length, and number of words per frame for the serial data. The SSI\_STCCR register is dedicated to the transmit section, and the SSI\_SRCCR register is dedicated to the receive section except in Synchronous mode, in which the SSI\_STCCR register controls both the receive and transmit sections. Power-on reset clears all SSI\_STCCR and SSI\_SRCCR bits. SSI reset does not affect the SSI\_STCCR and SSI\_SRCCR bits. The control bits are described in the following paragraphs. Although the bit patterns of the SSI\_STCCR and SSI\_SRCCR registers are the same, the contents of these two registers can be programmed differently.

**Table 48-54. SSI Data Length**

WL3	WL2	WL1	WL0	Number of Bits/Word	Supported in Implementation
0	0	0	0	2	No
0	0	0	1	4	No
0	0	1	0	6	No
0	0	1	1	8	Yes
0	1	0	0	10	Yes
0	1	0	1	12	Yes
0	1	1	0	14	No
0	1	1	1	16	Yes
1	0	0	0	18	Yes
1	0	0	1	20	Yes
1	0	1	0	22	Yes
1	0	1	1	24	Yes
1	1	0	0	26	No
1	1	0	1	28	No
1	1	1	0	30	No
1	1	1	1	32	No

Address: Base address + 28h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0													DIV2	PSR	WL3_WL0
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	WL3_WL0			DC4_DC0					PM7_PM0							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SSIx\_SRCR field descriptions

Field	Description
31–19 Reserved	This read-only field is reserved and always has the value 0.
18 DIV2	Divide By 2. This bit controls a divide-by-two divider in series with the rest of the prescalers. 0 Divider bypassed. 1 Divider used to divide clock by 2.
17 PSR	Prescaler Range. This bit controls a fixed divide-by-eight prescaler in series with the variable prescaler. It extends the range of the prescaler for those cases where a slower bit clock is required. 0 Prescaler bypassed. 1 Prescaler used to divide clock by 8.
16–13 WL3_WL0	Word Length Control. These bits are used to control the length of the data words being transferred by the SSI. These bits control the Word Length Divider in the Clock Generator. They also control the frame sync pulse length when the FSL bit is cleared. In I2S Master mode, the SSI works with a fixed word length of 32, and the WL bits are used to control the amount of valid data in those 32 bits. In AC97 Mode of operation, if word length is set to any value other than 16 bits, it will result in a word length of 20 bits.
12–8 DC4_DC0	Frame Rate Divider Control. These bits are used to control the divide ratio for the programmable frame rate dividers. The divide ratio works on the word clock. In Normal mode, this ratio determines the word transfer rate. In Network mode, this ratio sets the number of words per frame. The divide ratio ranges from 1 to 32 in Normal mode and from 2 to 32 in Network mode.  In Normal mode, a divide ratio of 1 (DC=00000) provides continuous periodic data word transfer. A bit-length frame sync must be used in this case.  These bits can be programmed with values ranging from "00000" to "11111" to control the number of words in a frame.
7–0 PM7_PM0	Prescaler Modulus Select. These bits control the prescale divider in the clock generator. This prescaler is used only in Internal Clock mode to divide the internal clock. The bit clock output is available at the clock port.  A divide ratio from 1 to 256 (PM[7:0] = 0x00 to 0xFF) can be selected. Refer to <a href="#">DIV2, PSR and PM Bit Description</a> for details regarding settings.

## 48.9.11 SSI FIFO Control/Status Register (SSIx\_SFCSR)

The SSI FIFO Control / Status Register indicates the status of the Transmit FIFO Empty flag, with different settings of the Transmit FIFO WaterMark bits and varying amounts of data in the Tx FIFO

.

**Table 48-56. Status of Transmit FIFO Empty Flag**

Transmit FIFO Watermark (TFWM)	Number of data in Tx-Fifo														
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
2	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
3	1	1	1	1	1	1	1	1	1	1	1	1	0	0	0
4	1	1	1	1	1	1	1	1	1	1	1	0	0	0	0
5	1	1	1	1	1	1	1	1	1	1	0	0	0	0	0
6	1	1	1	1	1	1	1	1	1	0	0	0	0	0	0
7	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0
8	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
9	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
10	1	1	1	1	1	0	0	0	0	0	0	0	0	0	0
11	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0
12	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
13	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
14	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Address: Base address + 2Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
R	RFCNT1								TFCNT1				RFWM1				TFWM1				RFCNT0				TFCNT0				RFWM0				TFWM0			
W																																				
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1				

**SSIx\_SFCSR field descriptions**

Field	Description
31–28 RFCNT1	Receive FIFO Counter1. These bits indicate the number of data words in Receive FIFO 1.  0000 0 data word in receive FIFO 0001 1 data word in receive FIFO 0010 2 data word in receive FIFO 0011 3 data word in receive FIFO 0100 4 data word in receive FIFO 0101 5 data word in receive FIFO 0110 6 data word in receive FIFO 0111 7 data word in receive FIFO 1000 8 data word in receive FIFO 1001 9 data word in receive FIFO 1010 10 data word in receive FIFO 1011 11 data word in receive FIFO 1100 12 data word in receive FIFO 1101 13 data word in receive FIFO

*Table continues on the next page...*

**SSIx\_SFCSR field descriptions (continued)**

Field	Description
	1110 14 data word in receive FIFO 1111 15 data word in receive FIFO
27–24 TFCNT1	Transmit FIFO Counter1. These bits indicate the number of data words in Transmit FIFO.  0000 0 data word in transmit FIFO 0001 1 data word in transmit FIFO 0010 2 data word in transmit FIFO 0011 3 data word in transmit FIFO 0100 4 data word in transmit FIFO 0101 5 data word in transmit FIFO 0110 6 data word in transmit FIFO 0111 7 data word in transmit FIFO 1000 8 data word in transmit FIFO 1001 9 data word in transmit FIFO 1010 10 data word in transmit FIFO 1011 11 data word in transmit FIFO 1100 12 data word in transmit FIFO 1101 13 data word in transmit FIFO 1110 14 data word in transmit FIFO 1111 15 data word in transmit FIFO
23–20 RFWM1	Receive FIFO Full WaterMark 1. These bits control the threshold at which the RFF1 flag will be set. The RFF1 flag is set whenever the data level in Rx FIFO 1 reaches the selected threshold.  0000 Reserved 0001 RFF set when at least one data word has been written to the Receive FIFO. Set when RxFIFO = 1,2.....15 data words 0010 RFF set when 2 or more data words have been written to the Receive FIFO. Set when RxFIFO = 2,3.....15 data words 0011 RFF set when 3 or more data words have been written to the Receive FIFO. Set when RxFIFO = 3,4.....15 data words 0100 RFF set when 4 or more data words have been written to the Receive FIFO. Set when RxFIFO = 4,5.....15 data words 0101 RFF set when 5 or more data words have been written to the Receive FIFO. Set when RxFIFO = 5,6.....15 data words 0110 RFF set when 6 or more data words have been written to the Receive.. Set when RxFIFO = 6,7.....15 data words 0111 RFF set when 7 or more data words have been written to the Receive FIFO. Set when RxFIFO = 7,8.....15 data words 1000 RFF set when 8 or more data words have been written to the Receive FIFO. Set when RxFIFO = 8,9.....15 data words 1001 RFF set when 9 or more data words have been written to the Receive FIFO. Set when RxFIFO = 9,10.....15 data words 1010 RFF set when 10 or more data words have been written to the Receive FIFO. Set when RxFIFO = 10,11.....15 data words 1011 RFF set when 11 or more data words have been written to the Receive FIFO. Set when RxFIFO = 11,12.....15 data words 1100 RFF set when 12 or more data words have been written to the Receive FIFO. Set when RxFIFO = 12,13.....15 data words

*Table continues on the next page...*



**SSIx\_SFCSR field descriptions (continued)**

<b>Field</b>	<b>Description</b>
	<p>1101 RFF set when 13 or more data words have been written to the Receive FIFO. Set when RxFIFO = 13,14,15 data words</p> <p>1110 RFF set when 14 or more data words have been written to the Receive FIFO. Set when RxFIFO = 14,15 data words</p> <p>1111 RFF set when 15 data words have been written to the Receive FIFO (default). Set when RxFIFO = 15 data words</p>
19–16 TFWM1	<p>Transmit FIFO Empty WaterMark 1. These bits control the threshold at which the TFE1 flag will be set. The TFE1 flag is set whenever the empty slots in Tx FIFO exceed or are equal to the selected threshold.</p> <p>0000 Reserved</p> <p>0001 TFE set when there are more than or equal to 1 empty slots in Transmit FIFO (default). Transmit FIFO empty is set when TxFIFO ≤ 14 data.</p> <p>0010 TFE set when there are more than or equal to 2 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 13 data.</p> <p>0011 TFE set when there are more than or equal to 3 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 12 data.</p> <p>0100 TFE set when there are more than or equal to 4 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 11 data.</p> <p>0101 TFE set when there are more than or equal to 5 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 10 data.</p> <p>0110 TFE set when there are more than or equal to 6 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 9 data.</p> <p>0111 TFE set when there are more than or equal to 7 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 8 data.</p> <p>1000 TFE set when there are more than or equal to 8 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 7 data.</p> <p>1001 TFE set when there are more than or equal to 9 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 6 data.</p> <p>1010 TFE set when there are more than or equal to 10 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 5 data.</p> <p>1011 TFE set when there are more than or equal to 11 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 4 data.</p> <p>1100 TFE set when there are more than or equal to 12 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 3 data.</p> <p>1101 TFE set when there are more than or equal to 13 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 2 data.</p> <p>1110 TFE set when there are more than or equal to 14 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO ≤ 1 data.</p> <p>1111 TFE set when there are 15 empty slots in Transmit FIFO. Transmit FIFO empty is set when TxFIFO = 0 data.</p>
15–12 RFCNT0	Receive FIFO Counter 0. These bits indicate the number of data words in Receive FIFO 0. See SSI_SFCSR_bf1 for details regarding settings for receive FIFO counter bits.
11–8 TFCNT0	Transmit FIFO Counter 0. These bits indicate the number of data words in Transmit FIFO 0. See SSI_SFCSR_bf2 for details regarding settings for transmit FIFO counter bits.
7–4 RFWM0	Receive FIFO Full WaterMark 0. These bits control the threshold at which the RFF0 flag will be set. The RFF0 flag is set whenever the data level in Rx FIFO 0 reaches the selected threshold. See SSI_SFCSR_bf3 for details regarding settings for receive FIFO watermark bits.
3–0 TFWM0	Transmit FIFO Empty WaterMark 0. These bits control the threshold at which the TFE0 flag will be set. The TFE0 flag is set whenever the empty slots in Tx FIFO exceed or are equal to the selected threshold. See SSI_SFCSR_bf4 for details regarding settings for transmit FIFO watermark bits.

## 48.9.12 SSI AC97 Control Register (SSIx\_SACNT)

Address: Base address + 38h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0					FRDIV						WR	RD	TIF	FV	AC97EN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SSIx\_SACNT field descriptions

Field	Description
31–11 Reserved	This read-only field is reserved and always has the value 0.
10–5 FRDIV	<p>Frame Rate Divider. These bits control the frequency of AC97 data transmission/reception. They are programmed with the number of frames for which the SSI should be idle, after operating in one frame. Through these bits, AC97 frequency of operation, from 48 KHz (000000) to 1 KHz (101111) can be achieved.</p> <p>Sample Value: 001010 (10 Decimal) = SSI will operate once every 11 frames.</p>
4 WR	<p>Write Command. This bit specifies whether the next frame will carry an AC97 Write Command or not. The programmer should take care that only one of the bits (WR or RD) is set at a time. When this bit is set, the corresponding tag bits (corresponding to Command Address and Command Data slots of the next Tx frame) are automatically set. This bit is automatically cleared by the SSI after completing transmission of a frame.</p> <p>0 Next frame will not have a Write Command. 1 Next frame will have a Write Command.</p>
3 RD	<p>Read Command. This bit specifies whether the next frame will carry an AC97 Read Command or not. The programmer should take care that only one of the bits (WR or RD) is set at a time. When this bit is set, the corresponding tag bit (corresponding to Command Address slot of the next Tx frame) is automatically set. This bit is automatically cleared by the SSI after completing transmission of a frame.</p> <p>0 Next frame will not have a Read Command. 1 Next frame will have a Read Command.</p>
2 TIF	<p>Tag in FIFO. This bit controls the destination of the information received in AC97 tag slot (Slot #0).</p> <p>0 <b>SATAG_REGISTER</b> — Tag info stored in SATAG register. 1 <b>RX_FIFO0</b> — Tag info stored in Rx FIFO 0.</p>
1 FV	Fixed/Variable Operation. This bit selects whether the SSI is in AC97 Fixed mode or AC97 Variable mode.

Table continues on the next page...

**SSIx\_SACNT field descriptions (continued)**

Field	Description
	0 <b>FIXED</b> — AC97 Fixed Mode. 1 <b>VARIABLE</b> — AC97 Variable Mode.
0 AC97EN	AC97 Mode Enable. This bit is used to enable SSI AC97 operation. Refer to <a href="#">AC97 Mode</a> for details of AC97 operation.  0 AC97 mode disabled. 1 SSI in AC97 mode.

**48.9.13 SSI AC97 Command Address Register (SSIx\_SACADD)**

Address: Base address + 3Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0													SACADD																		
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SSIx\_SACADD field descriptions**

Field	Description
31–19 Reserved	This read-only field is reserved and always has the value 0.
18–0 SACADD	AC97 Command Address. These bits store the Command Address Slot information (bit 19 of the slot is sent in accordance with the Read and Write Command bits in SSI_SACNT register). These bits can be updated by a direct write from the Core. They are also updated with the information received in the incoming Command Address Slot. If the contents of these bits change due to an update, the CMDAU bit in SISR is set.

**48.9.14 SSI AC97 Command Data Register (SSIx\_SACDAT)**

Address: Base address + 40h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0													SACDAT																		
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**SSIx\_SACDAT field descriptions**

Field	Description
31–20 Reserved	This read-only field is reserved and always has the value 0.
19–0 SACDAT	AC97 Command Data. The outgoing Command Data Slot carries the information contained in these bits. These bits can be updated by a direct write from the Core. They are also updated with the information received in the incoming Command Data Slot. If the contents of these bits change due to an update, the

*Table continues on the next page...*

**SSIx\_SACDAT field descriptions (continued)**

Field	Description
	CMDDU bit in SISR is set. These bits are transmitted only during AC97 Write Command. During AC97 Read Command, 0x00000 is transmitted in time slot #2.

**48.9.15 SSI AC97 Tag Register (SSIx\_SATAG)**

Address: Base address + 44h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																SATAG															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SSIx\_SATAG field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–0 SATAG	AC97 Tag Value. Writing to this register (by the Core) sets the value of the Tx-Tag in AC97 fixed mode of operation. On a read, the Core gets the Rx-Tag Value received (in the last frame) from the Codec. If TIF bit in SSI_SACNT register is set, the TAG value is also stored in Rx-FIFO in addition to SATAG register. When the received Tag value changes, the RXT bit in SISR register is set.  Bits SATAG[1:0] convey the Codec -ID. In current implementation only Primary Codecs are supported. Thus writing value 2'b00 to this field is mandatory.

**48.9.16 SSI Transmit Time Slot Mask Register (SSIx\_STMSK)**

Address: Base address + 48h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	STMSK																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**SSIx\_STMSK field descriptions**

Field	Description
31–0 STMSK	Transmit Mask. These bits indicate which slot has been masked in the current frame. The Core can write to this register to control the time slots in which the SSI transmits data. Each bit has info corresponding to the respective time slot in the frame. Transmit mask bits should not be used in I2S Slave mode of operation. SSI_STMSK register value must be set before enabling Transmission.  0 Valid Time Slot. 1 Time Slot masked (no data transmitted in this time slot).

## 48.9.17 SSI Receive Time Slot Mask Register (SSIx\_SRMSK)

Address: Base address + 4Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SRMSK																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SSIx\_SRMSK field descriptions

Field	Description
31–0 SRMSK	Receive Mask. These bits indicate which slot has been masked in the current frame. The Core can write to this register to control the time slots in which the SSI receives data. Each bit has info corresponding to the respective time slot in the frame. SSI_SRMSK register value must be set before enabling Receiver. Receive mask bits should not be used in I2S Slave mode of operation.  0 Valid Time Slot. 1 Time Slot masked (no data received in this time slot).

## 48.9.18 SSI AC97 Channel Status Register (SSIx\_SACCST)

Address: Base address + 50h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																SACCST															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SSIx\_SACCST field descriptions

Field	Description
31–10 Reserved	This read-only field is reserved and always has the value 0.
9–0 SACCST	AC97 Channel Status. These bits indicate which data slot has been enabled in AC97 variable mode operation. This register is updated in case the core enables/disables a channel through a write to SSI_SACCEN/SSI_SACCDIS register or the external codec enables a channel by sending a '1' in the corresponding SLOTREQ bit. Bit [0] corresponds to the first data slot in an AC97 frame (Slot #3) and Bit [9] corresponds to the tenth data slot (slot #12). The contents of this register only have relevance while the SSI is operating in AC97 variable mode. Writes to this register result in an error response on the block interface.  0 Data channel disabled. 1 Data channel enabled.

## 48.9.19 SSI AC97 Channel Enable Register (SSIx\_SACCEN)

Address: Base address + 54h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																															
W																	SACCEN															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SSIx\_SACCEN field descriptions

Field	Description
31–10 Reserved	This read-only field is reserved and always has the value 0.
9–0 SACCEN	AC97 Channel Enable. The Core writes a '1' to these bits to enable an AC97 data channel. Writing a '0' has no effect. Bit [0] corresponds to the first data slot in an AC97 frame (Slot #3) and Bit [9] corresponds to the tenth data slot (slot #12). Writes to these bits only have effect in the AC97 Variable mode of operation. These bits are always read as '0' by the Core.  0 Write Has no effect. 1 Write Enables the corresponding data channel.

## 48.9.20 SSI AC97 Channel Disable Register (SSIx\_SACCDIS)

Address: Base address + 58h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																															
W																	SACCDIS															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### SSIx\_SACCDIS field descriptions

Field	Description
31–10 Reserved	This read-only field is reserved and always has the value 0.
9–0 SACCDIS	AC97 Channel Disable. The Core writes a '1' to these bits to disable an AC97 data channel. Writing a '0' has no effect. Bit [0] corresponds to the first data slot in an AC97 frame (Slot #3) and Bit [9] corresponds to the tenth data slot (slot #12). Writes to these bits only have effect in the AC97 Variable mode of operation. These bits are always read as '0' by the Core.  0 Write Has no effect. 1 Write Disables the corresponding data channel.

## Chapter 49

# Temperature Monitor (TEMPMON)

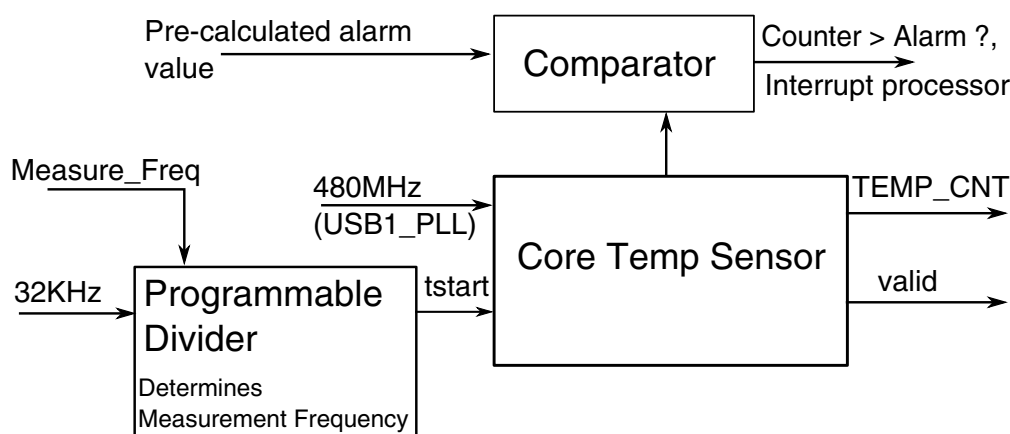
### 49.1 Overview

The temperature sensor module implements a temperature sensor/conversion function based on a temperature-dependent voltage to time conversion.

The module features an alarm function that can raise an interrupt signal if the temperature is above a specified threshold. A self-repeating mode can also be programmed which executes a temperature sensing operation based on a programmed delay.

Software can use this module to monitor the on-die temperature and take appropriate actions such as throttling back the core frequency when a temperature interrupt is set.

The high-level implementation of the temperature sensor is shown in the figure below.



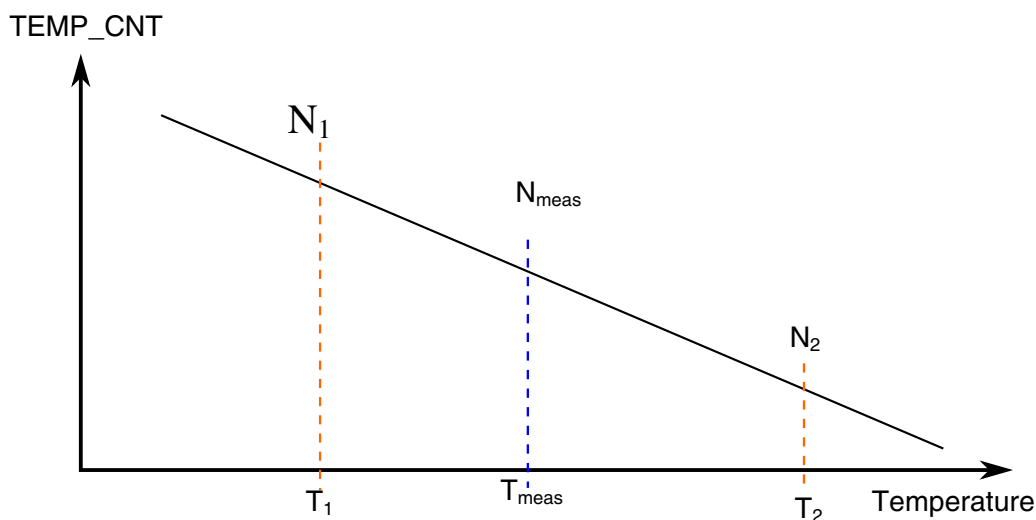
**Figure 49-1. High Level Temp Sensor System Diagram**

As shown in the figure above, the temperature sensor uses and assumes that the bandgap reference, 480MHz PLL and 32KHz RTC modules are properly programmed and fully settled for correct operation.

## 49.2 Software Usage Guidelines

During normal system operation software can use the temperature sensor counter output (TEMP\_CNT) in conjunction with the fused temperature calibration data to determine the on-die operational temperature or to set an over-temperature interrupt alarm to within a couple of °C.

Based on calibration, two sets of temperature and counter values will be available via fuses on the device. These data points will correspond to the points ( $N_1, T_1$ ) and ( $N_2, T_2$ ) in the curve below.



**Figure 49-2. Temperature Measurement Cycle**

After a temperature measurement cycle, software should use the calibration points in conjunction with the temperature code value in the `TEMPMON_TEMPSENSE0[TEMP_CNT]` bitfield to calculate the temperature for the device using the following equation:

$$T_{\text{meas}} = T_2 - (N_{\text{meas}} - N_2) * ((T_2 - T_1) / (N_1 - N_2))$$

Likewise, to determine the alarm counter value to be written in the `TEMPMON_TEMPSENSE0` register for a temperature based interrupt, the above equation can be solved for the  $N_{\text{meas}}$  value that should be used based on the desired temperature trigger.

The temperature calibration point fuse values are available in the `OCOTP_ANA1` register. The temperature calibration values are fused individually for each part in the product testing process. The fields of this register are described in the following table.



**Table 49-1. OCOTP\_ANA1 Temperature Sensor Calibration Data**

Bit Range	Bit Mask	Name	Description
[31:20]	FFF0_0000h	ROOM_COUNT	Value of TEMPMON_TEMPSENSE0[TEMP_VALUE] after a measurement cycle at room temperature (25.0 °C).
[19:8]	000F_FF00h	HOT_COUNT	Value of TEMPMON_TEMPSENSE0[TEMP_VALUE] after a measurement cycle at the hot temperature, i.e. HOT_TEMP.
[7:0]	0000_00FFh	HOT_TEMP	The hot temperature test point. Each LSB equals 1 °C.

The points on the calibration curve are as follows.

- $(N_1, T_1) = (\text{ROOM\_COUNT}, 25.0)$
- $(N_2, T_2) = (\text{HOT\_COUNT}, \text{HOT\_TEMP})$
- $(N_{\text{meas}}, T_{\text{meas}}) = (\text{TEMP\_CNT}, T_{\text{meas}})$

Substituting the fields from OCOTP\_ANA1 into the earlier equation results in the following:

$$T_{\text{meas}} = \text{HOT\_TEMP} - (N_{\text{meas}} - \text{HOT\_COUNT}) * ((\text{HOT\_TEMP} - 25.0) / (\text{ROOM\_COUNT} - \text{HOT\_COUNT}))$$

## 49.3 TEMPMON Memory Map/Register Definition

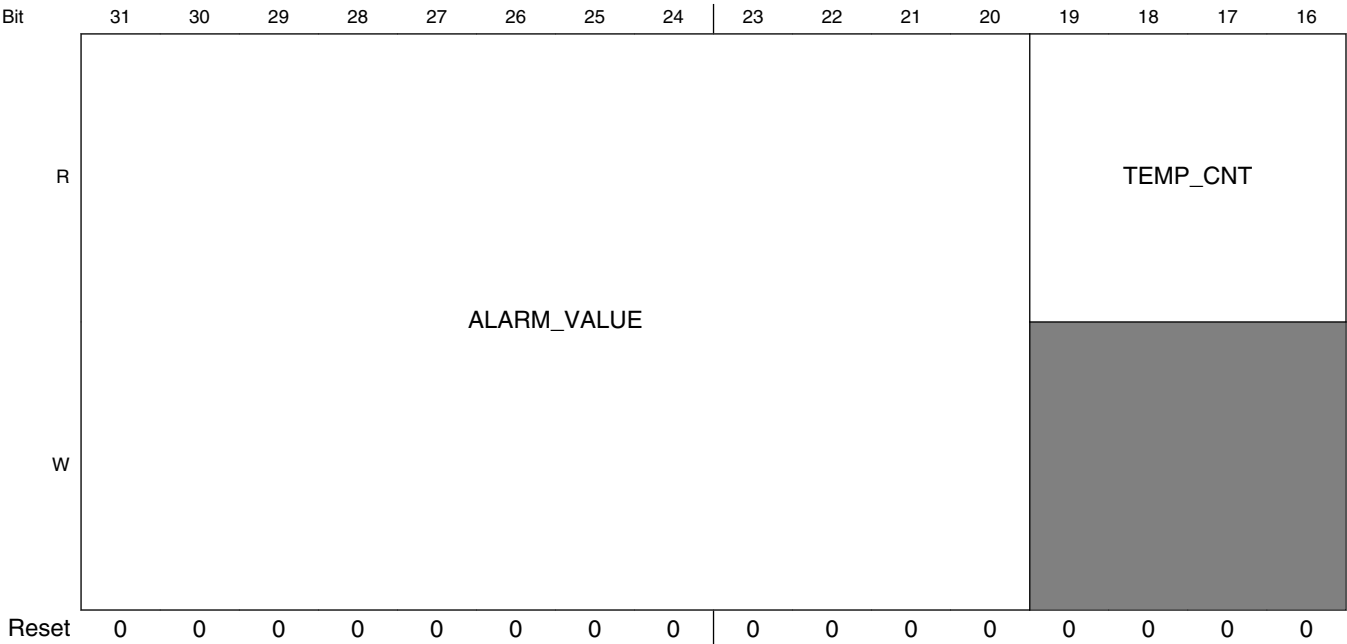
### TEMPMON memory map

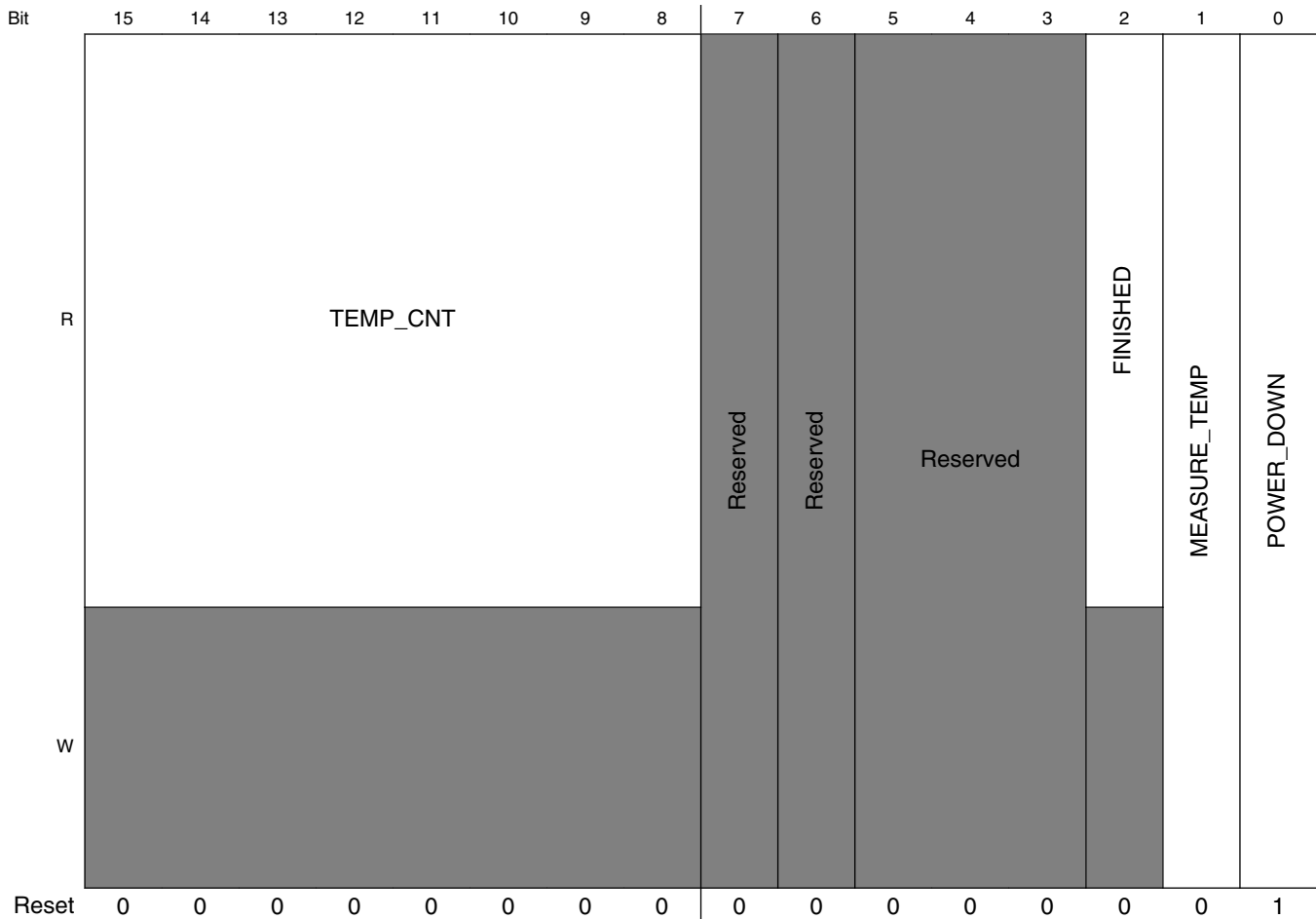
Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20C_8180	Tempsensor Control Register 0 (TEMPMON_TEMPSENSE0)	32	R/W	0000_0001h	<a href="#">49.3.1/2918</a>
20C_8184	Tempsensor Control Register 0 (TEMPMON_TEMPSENSE0_SET)	32	R/W	0000_0001h	<a href="#">49.3.1/2918</a>
20C_8188	Tempsensor Control Register 0 (TEMPMON_TEMPSENSE0_CLR)	32	R/W	0000_0001h	<a href="#">49.3.1/2918</a>
20C_818C	Tempsensor Control Register 0 (TEMPMON_TEMPSENSE0_TOG)	32	R/W	0000_0001h	<a href="#">49.3.1/2918</a>
20C_8190	Tempsensor Control Register 1 (TEMPMON_TEMPSENSE1)	32	R/W	0000_0001h	<a href="#">49.3.2/2920</a>
20C_8194	Tempsensor Control Register 1 (TEMPMON_TEMPSENSE1_SET)	32	R/W	0000_0001h	<a href="#">49.3.2/2920</a>
20C_8198	Tempsensor Control Register 1 (TEMPMON_TEMPSENSE1_CLR)	32	R/W	0000_0001h	<a href="#">49.3.2/2920</a>
20C_819C	Tempsensor Control Register 1 (TEMPMON_TEMPSENSE1_TOG)	32	R/W	0000_0001h	<a href="#">49.3.2/2920</a>

### 49.3.1 Tempsensor Control Register 0 (TEMPMON\_TEMPSENSE0n)

This register defines the basic controls for the temperature sensor minus the frequency of automatic sampling which is defined in the tempsensor.

Address: 20C\_8000h base + 180h offset + (4d × i), where i=0d to 3d





TEMPMON\_TEMPSENSE0n field descriptions

Field	Description
31–20 ALARM_VALUE	This bit field contains the temperature count (raw sensor output) that will generate an alarm interrupt.
19–8 TEMP_CNT	This bit field contains the last measured temperature count.
7 -	This field is reserved. Reserved.
6 -	This field is reserved. Reserved.
5–3 -	This field is reserved. Reserved
2 FINISHED	Indicates that the latest temp is valid. This bit should be cleared by the sensor after the start of each measurement.  0 <b>INVALID</b> — Last measurement is not ready yet. 1 <b>VALID</b> — Last measurement is valid.
1 MEASURE_TEMP	Starts the measurement process. If the measurement frequency is zero in the TEMPSENSE1 register, this results in a single conversion.

Table continues on the next page...

**TEMPMON\_TEMPSENSE0n field descriptions (continued)**

Field	Description
	0 <b>STOP</b> — Do not start the measurement process. 1 <b>START</b> — Start the measurement process.
0 POWER_DOWN	This bit powers down the temperature sensor. 0 <b>POWER_UP</b> — Enable power to the temperature sensor. 1 <b>POWER_DOWN</b> — Power down the temperature sensor.

**49.3.2 Tempsensor Control Register 1  
(TEMPMON\_TEMPSENSE1n)**

This register defines the automatic repeat time of the temperature sensor.

Address: 20C\_8000h base + 190h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	Reserved																MEASURE_FREQ																
W	Reserved																MEASURE_FREQ																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1		

**TEMPMON\_TEMPSENSE1n field descriptions**

Field	Description
31–16 -	This field is reserved. Reserved.
15–0 MEASURE_ FREQ	This bits determines how many RTC clocks to wait before automatically repeating a temperature measurement. The pause time before remeasuring is the field value multiplied by the RTC period.  0x0000 Defines a single measurement with no repeat. 0x0001 Updates the temperature value at a RTC clock rate. 0x0002 Updates the temperature value at a RTC/2 clock rate. ... — 0xFFFF Determines a two second sample period with a 32.768KHz RTC clock. Exact timings depend on the accuracy of the RTC clock.

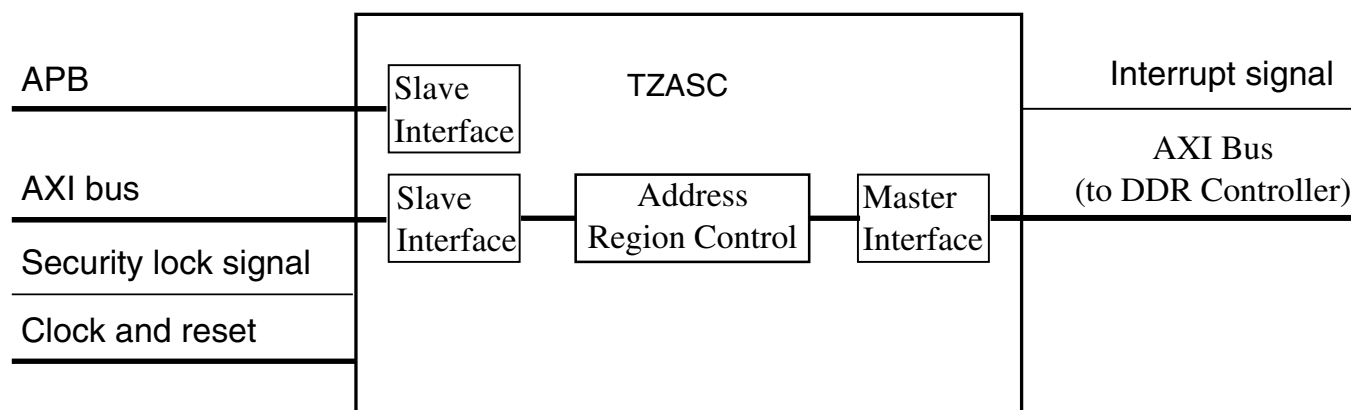
## Chapter 50

# TrustZone Address Space Controller (TZASC)

### 50.1 Overview

The TrustZone Address Space Controller (TZASC) protects security-sensitive SW and data in a trusted execution environment against potentially compromised SW running on the platform.

The TZASC block diagram is shown in figure below.



**Figure 50-1. TZASC Block Diagram**

The TZASC is an IP by ARM ("CoreLink™ TrustZone Address Space Controller TZC-380"), designed to provide configurable protection over program (SW) memory space.

The main features of TZASC are:

- Supports 16 independent address regions
- Access controls are independently programmable for each address region
- Sensitive registers may be locked
- Host interrupt may be programmed to signal attempted access control violations

- AXI master/slave interfaces for transactions
- APB slave interface for configuration and status reporting

## 50.2 Clocks

The table found here describes the clock sources for TZASC.

Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 50-1. TZASC Clocks**

Clock name	Clock Root	Description
ackl	mmdc_axi_clk_root	Module clock

## 50.3 i.MX 6SoloLite Specific Configuration

The i.MX 6SoloLite uses a single TZASC instance, on the DDR controller (MMDC), to provide address protection based on access security level.

The i.MX 6SoloLite utilizes bus muxing logic to route DDR memory traffic through or bypass the TZASC module. By default, the TZASC is bypassed, and its clocks are gated off.

Enabling TZASC is expected to have a slight impact on memory performance. Exact value cannot be stated, since varies, depending on specific application software.

The proper and preferred method of enabling the TZASC, is by burning the TZASC\_ENABLE fuse

For every power-up cycle, with TZASC\_ENABLE fuse burned, the Boot ROM code will seamlessly handle the enable and engage of the TZASC module and its clocks, leaving it in an active state, (i.e. enabled, and not bypassed). From this state on, it is the responsibility of OS image, to configure the memory regions protection, per a specific application / use-case needs.

A configuration lock bit ("TZASC\_BOOT\_LOCK" in GPR3 register of IOMUXC) once set, will block any attempts to change the security settings, past the OS image configuration code settings. The configuration locking is in place, until the next hardware reset cycle.

**NOTE**

Engaging the TZASC functionality (i.e. - not bypassed), has to be done while DDR bus is guaranteed to be idle, with no pending transactions.

Enabling TZASC, w/o use of the associated fuse, is possible, but not trivial, since has to be done while traffic to DDR is guaranteed to be stopped. A typical way of achieving this, is by:

- Ensuring no other master can issue accesses to DDR.
- Protect against program flow change, to DDR space, (Disable interrupts, ans such).
- Run switching code from internal RAM.

TZASC enabling code has to handle the data-path mux control (via TZASC\_BYPASS bits in GPR9, in IOMUXC module) as well as clock enabling (in LPCG module), and configuration locking, if desired (via set of TZASC\_BOOT\_LOCK bit in GPR3, IOMUXC module).

The TZASC\_BYPASS bit in GPR9 register, once set, preserve its value until the next power-up cycle ("Sticky" type), in order to protect against unauthorized 'disable' operation.

## 50.4 Address Mapping in various memory mapping modes

The address configured to the TZASC controller(s) must match the "local addresses" as being passed on to the DDR controller(s).

The DDR controller "local addresses" are referred to, as the addresses seen by the DDR controller. In the single channel, single controller scheme of i.MX 6SoloLite, the local addresses are in the 2-4GB range.

Memory "aliasing" implications on TZASC settings - in systems which does not utilize the maximal supported DDR space the controller is designed for, the whole DDR memory map becomes "aliased" (replicated) by the size of the physical memory used. In such cases, the TZASC must be configured to protect all aliased regions as well (i.e. effectively reducing the number of available TZASC regions, since all aliased regions must be handled, for each "real" space needing protection).

For complete details on TZASC functionality and the programming model, see the ARM document, "CoreLink™ TrustZone Address Space Controller TZC-380 Technical Reference Manual, (Rev r0p1 or newer)", available at <http://infocenter.arm.com>.





# Chapter 51

## Universal Asynchronous Receiver/Transmitter (UART)

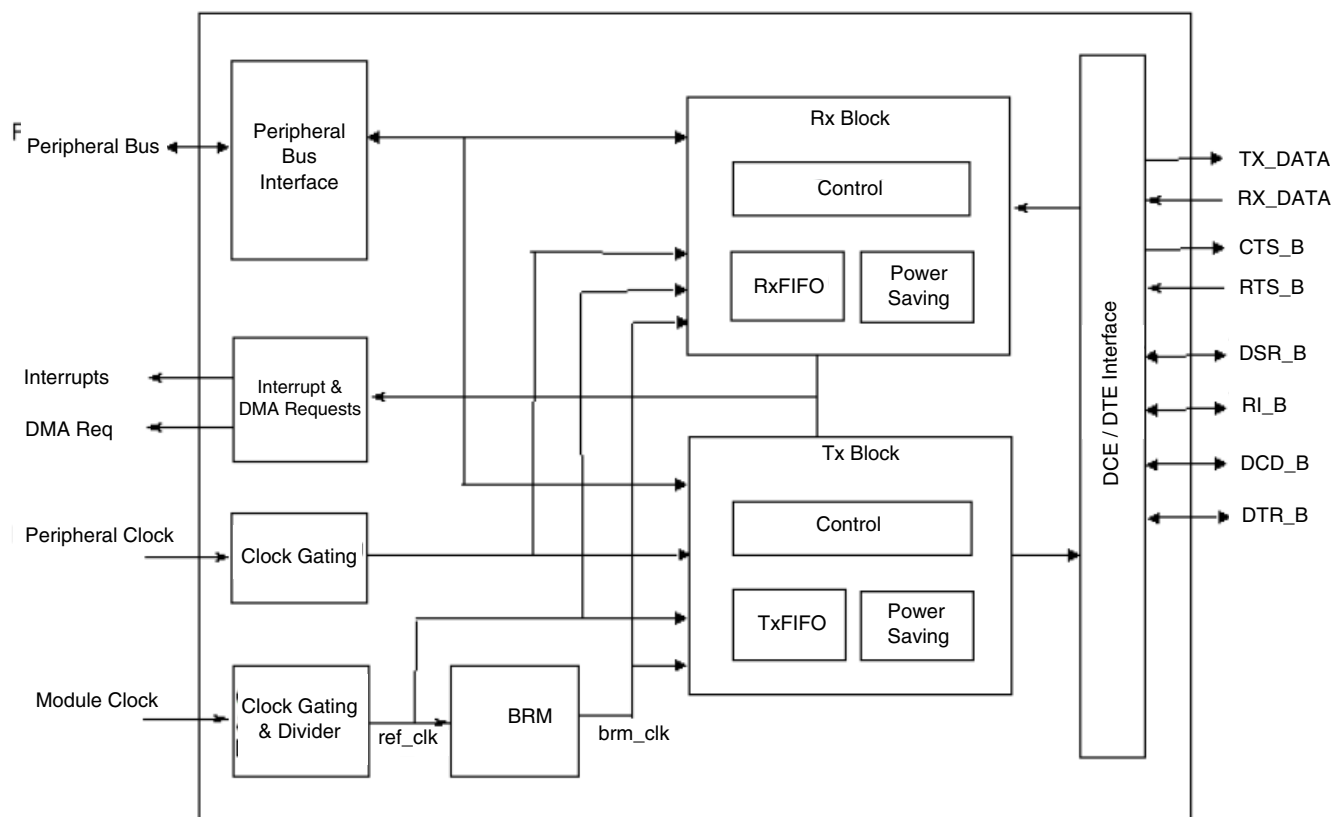
### 51.1 Overview

Universal Asynchronous Receiver/Transmitter (UART) provides serial communication capability with external devices through a level converter and an RS-232 cable or through use of external circuitry that converts infrared signals to electrical signals (for reception) or transforms electrical signals to signals that drive an infrared LED (for transmission) to provide low speed IrDA compatibility.

UART supports NRZ encoding format , RS485 compatible 9 bit data format and IrDA-compatible infrared slow data rate (SIR) format.

The following figure is the UART block diagram.

The "Module Clock" is the UART\_CLK which comes from CCM. The "Peripheral Clock" is the IPG\_CLK which comes from CCM.



**Figure 51-1. UART Block Diagram**

### 51.1.1 Features

The UART includes the following features:

- High-speed TIA/EIA-232-F compatible, up to Mbit/s
- Serial IR interface low-speed, IrDA-compatible (up to 115.2 Kbit/s)
- 9-bit or Multidrop mode (RS-485) support (automatic slave address detection)
- 7 or 8 data bits for RS-232 characters, or 9 bit RS-485 format
- 1 or 2 stop bits
- Programmable parity (even, odd, and no parity)
- Hardware flow control support for request to send (RTS\_B) and clear to send (CTS\_B) signals
- RS-485 driver direction control via CTS\_B signal
- Edge-selectable RTS\_B and edge-detect interrupts
- Status flags for various flow control and FIFO states
- Voting logic for improved noise immunity (16x oversampling)
- Transmitter FIFO empty interrupt suppression
- UART internal clocks enable/disable

- Auto baud rate detection (up to 115.2 Kbit/s)
- Receiver and transmitter enable/disable for power saving
- RX\_DATA input and TX\_DATA output can be inverted respectively in RS-232/RS-485 mode
- DCE/DTE capability
- RTS\_B, IrDA asynchronous wake (AIRINT), receive asynchronous wake (AWAKE), RI\_B (DTE only), DCD\_B (DTE only), DTR\_B (DCE only) and DSR\_B (DTE only) interrupts wake the processor from STOP mode
- Maskable interrupts
- Two DMA Requests (TxFIFO DMA Request and RxFIFO DMA Request)
- Escape character sequence detection
- Software reset (SRST\_B)
- Two independent, 32-entry FIFOs for transmit and receive
- The peripheral clock can be totally asynchronous with the module clock. The module clock determines baud rate. This allows frequency scaling on peripheral clock (such as during DVFS mode) while remaining the module clock frequency and baud rate.

### 51.1.2 Modes of Operation

- Serial RS-232NRZ mode
- 9-bit RS-485 mode
- IrDA mode

To set UART in different modes, see the table below for reference.

**Table 51-1. UART Mode Defination**

MDEN (UMCR[0])	IREN (UCR1[7])	UART Mode	Description
0	0	RS-232	RXD/TXD data is serial RS-232 NRZ format
0	1	IrDA (Interface)	RXD/TXD data is IrDA-compatible infrared slow data rate (SIR) format
1	0	RS-485	RXD/TXD data is RS485 compatible 9 bit data format
1	1	Undefined	Undefined

## 51.2 External Signals

The following table describes the external signals of UART:

**Table 51-2. UART1 External Signals**

Signal	Description	Pad	Mode	Direction
UART1_CTS_B	Clear to send	I2C1_SDA	ALT1	O
UART1_RTS_B	Request to send	I2C1_SCL	ALT1	I
UART1_RX_DATA	Serial / infrared data receive	UART1_RXD	ALT0	I
UART1_TX_DATA	Serial/infrared data transmit	UART1_TXD	ALT0	O

**Table 51-3. UART2 External Signals**

Signal	Description	Pad	Mode	Direction
UART2_CTS_B	Clear to send	EPDC_D15	ALT1	O
		LCD_RESET	ALT4	
		SD2_DAT7	ALT2	
UART2_RTS_B	Request to send	EPDC_D14	ALT1	I
		LCD_VSYNC	ALT4	
		SD2_DAT6	ALT2	
UART2_RX_DATA	Serial / infrared data receive	EPDC_D12	ALT1	I
		LCD_ENABLE	ALT4	
		SD2_DAT4	ALT2	
UART2_TX_DATA	Serial/infrared data transmit	EPDC_D13	ALT1	O
		LCD_HSYNC	ALT4	
		SD2_DAT5	ALT2	

**Table 51-4. UART3 External Signals**

Signal	Description	Pad	Mode	Direction
UART3_CTS_B	Clear to send	ECSPi2_SS0	ALT2	O
		EPDC_BDR1	ALT2	
UART3_RTS_B	Request to send	ECSPi2_MISO	ALT2	I
		EPDC_BDR0	ALT2	
UART3_RX_DATA	Serial / infrared data receive	AUD_RXFS	ALT2	I
		ECSPi2_SCLK	ALT2	
		EPDC_VCOM0	ALT2	
UART3_TX_DATA	Serial/infrared data transmit	AUD_RXC	ALT2	O
		ECSPi2_MOSI	ALT2	
		EPDC_VCOM1	ALT2	

**Table 51-5. UART4 External Signals**

Signal	Description	Pad	Mode	Direction
UART4_CTS_B	Clear to send	AUD_TXD	ALT2	O
		KEY_ROW7	ALT1	
		SD1_DAT7	ALT4	
UART4_RTS_B	Request to send	AUD_TXFS	ALT2	I
		KEY_COL7	ALT1	
		SD1_DAT6	ALT4	
UART4_RX_DATA	Serial / infrared data receive	AUD_RXD	ALT2	I
		KEY_COL6	ALT1	
		SD1_DAT4	ALT4	
		UART1_RXD	ALT2	
UART4_TX_DATA	Serial/infrared data transmit	AUD_TXC	ALT2	O
		KEY_ROW6	ALT1	
		SD1_DAT5	ALT4	
		UART1_TXD	ALT2	

## 51.2.1 Detailed Signal Descriptions

### 51.2.1.1 Serial/IrDA Signals

#### 51.2.1.1.1 RXD - Data Receive

Input asynchronous data receive in Serial and IrDA modes.

#### 51.2.1.1.2 TXD - Data Transmit

Output asynchronous data transmit in Serial and IrDA modes.

### 51.2.1.2 Modem Control Signals

#### 51.2.1.2.1 $\overline{\text{CTS}}$ - Clear To Send

Output in DCE mode. Input in DTE mode. This signal informs the remote modem that UART is ready to receive data.

#### 51.2.1.2.2 $\overline{\text{RTS}}$ - Request To Send

Input in DCE mode. Output in DTE mode. This signal informs UART that remote modem is ready to receive data.

#### 51.2.1.2.3 $\overline{\text{DSR}}$ - Data Set Ready

Input in DTE mode. Indicates to UART that remote modem is operational.

Output in DCE mode. Indicates to remote modem that UART is operational.

#### 51.2.1.2.4 $\overline{\text{DCD}}$ - Data Carrier Detected

Input in DTE mode. Indicates to UART that a good carrier is being received from the remote modem.

Output in DCE mode. Indicates to remote device that a good carrier is being received from the UART.

#### 51.2.1.2.5 $\overline{\text{DTR}}$ - Data Terminal Ready

Input in DCE mode. Indicates to UART (in DCE mode) that remote device (in DTE mode) is operational.

Output in DTE mode. Indicates to remote modem (in DCE mode) that UART (in DTE mode) is operational.

#### 51.2.1.2.6 $\overline{\text{RI}}$ - Ring Indicator

Input in DTE mode. Indicates to UART that remote modem is detecting a ringing tone.

Output in DCE mode. Indicates to remote device that UART is detecting a ringing tone.

### 51.2.1.3 Interrupt Signals

#### 51.2.1.3.1 *interrupt\_uart* - UART Interrupt

Output interrupt request.

### 51.2.1.4 DMA Request Signals

#### 51.2.1.4.1 *dma\_req\_rx* - Receiver DMA Request

Output DMA Request signal for receiver interface.

#### 51.2.1.4.2 *dma\_req\_tx* - Transmitter DMA Request

Output DMA Request signal for transmitter interface. Set at 0 when TXDMAEN (UCR1[3]) is at 1 and TRDY (USR1[13]) is also at 1.

### 51.2.1.5 Special Signals

#### 51.2.1.5.1 *stop\_req* - Stop Mode

Input stop mode. Indicates to UART that ARM platform is going to enter in Stop Mode and clocks are going to stop running.

See [Low Power Modes](#) for more information about Stop Mode.

#### 51.2.1.5.2 *doze\_req* - Doze Mode

Input doze mode. ARM platform requests UART to switch in doze mode (power saving mode).

See [Low Power Modes](#) for more information about Doze Mode.

#### 51.2.1.5.3 *debug\_req* - Debug Mode

Input debug mode. Indicates UART it has to enter in debug mode.

See [UART Operation in System Debug State](#), for more information about Debug Mode.

## 51.3 Clocks

The table found here describes the clock sources for UART.

Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 51-6. UART Clocks**

Clock name	Clock Root	Description
ipg_clk	ipg_clk_root	Peripheral clock

*Table continues on the next page...*

**Table 51-6. UART Clocks (continued)**

Clock name	Clock Root	Description
ipg_clk_s	ipg_clk_root	Peripheral access clock
ipg_perclk	uart_clk_root	Module clock

## 51.4 Functional Description

This section provides a complete functional description of the block.

### 51.4.1 Interrupts and DMA Requests

See the following table for the lists of all interrupt and DMA signals and associated interrupt and DMA sources of the UART. See register description section for explanation of interrupt/DMA enable and status.

**Table 51-7. Interrupts and DMA**

Interrupt/DMA Output	Interrupt/DMA Enable	Enable Register Location	Interrupt/DMA Flag	Flag Register Location
<i>interrupt_uart</i>	RRDYEN	UCR1 (bit 9)	RRDY	USR1 (bit 9)
	IDEN	UCR1 (bit 12)	IDLE	USR2 (bit 12)
	DREN	UCR4 (bit 0)	RDR	USR2 (bit 0)
	RXDSEN	UCR3 (bit 6)	RXDS	USR1 (bit 6)
	ATEN	UCR2 (bit 3)	AGTIM	USR1 (bit 8)
<i>interrupt_uart</i>	TXMPTYEN	UCR1 (bit 6)	TXFE	USR2 (bit 14)
	TRDYEN	UCR1 (bit 13)	TRDY	USR1 (bit 13)
	TCEN	UCR4 (bit 3)	TXDC	USR2 (bit 3)

*Table continues on the next page...*



**Table 51-7. Interrupts and DMA (continued)**

Interrupt/DMA Output	Interrupt/DMA Enable	Enable Register Location	Interrupt/DMA Flag	Flag Register Location
<i>interrupt_uart</i>	OREN	UCR4 (bit 1)	ORE	USR2 (bit 1)
	BKEN	UCR4 (bit 2)	BRCD	USR2 (bit 2)
	WKEN	UCR4 (bit 7)	WAKE	USR2 (bit 7)
	ADEN	UCR1 (bit 15)	ADET	USR2 (bit 15)
	ACIEN	UCR3 (bit 0)	ACST	USR2 (bit 11)
	ESCI	UCR2 (bit 15)	ESCF	USR1 (bit 11)
	ENIRI	UCR4 (bit 8)	IRINT	USR2 (bit 8)
	AIRINTEN	UCR3 (bit 5)	AIRINT	USR1 (bit 5)
	AWAKEN	UCR3 (bit 4)	AWAKE	USR1 (bit 4)
	FRAERREN	UCR3 (bit 11)	FRAERR	USR1 (bit 10)
	PARERREN	UCR3 (bit 12)	PARITYERR	USR1 (bit 15)
	RTSDEN	UCR1 (bit 5)	RTSD	USR1 (bit 12)
	RTSEN	UCR2 (bit 4)	RTSF	USR2 (bit 4)
	DTREN (DCE)	UCR3 (bit 13)	DTRF	USR2 (bit 13)
	RI (DTE)	UCR3 (bit 8)	RIDELT	USR2 (bit 10)
	DCD (DTE)	UCR3 (bit 9)	DCDDELT	USR2 (bit 6)
	DTRDEN	UCR3 (bit 3)	DTRD	USR1 (bit 7)
	SADEN	UMCR (bit 3)	SAD	USR1 (bit 3)
<i>dma_req_rx</i>	RXDMAEN	UCR1 (bit 8)	RRDY	USR1 (bit 9)
	ATDMAEN	UCR1 (bit 2)	AGTIM	USR1 (bit 8)
	IDDMAEN	UCR4 (bit 6)	IDLE	USR2 (bit 12)
<i>dma_req_tx</i>	TXDMAEN	UCR1 (bit 3)	TRDY	USR1 (bit 13)

## 51.4.2 Clocks

This section describes clocks and special clocking requirements of the UART.

### 51.4.2.1 Clock requirements

UART module receives 2 clocks, *peripheral\_clock* and *module\_clock*. The *peripheral\_clock* is used as write clock of the TxFIFO, read clock of the RxFIFO and synchronization of the modem control input pins. It must always be running when UART is enabled. There is an exception in stop mode (see [Clocking in Low-Power Modes](#)).

The *module\_clock* is for all the state machines, writing RxFIFO, reading TxFIFO, etc. It must always be running when UART is sending or receiving characters. This clock is used in order to allow frequency scaling on *peripheral\_clock* without changing configuration of baud rate (*module\_clock* staying at a fixed frequency).

The constraints on *peripheral\_clock* and *module\_clock* are as follows:

- *peripheral\_clock* and *module\_clock* can totally be asynchronous. Of course, they can also be synchronous.
- Due to the 16x oversampling of the incoming characters, *module\_clock* frequency must always be greater or equal to 16x the maximum baud rate. For example, if max baud rate is 4 Mbit/s, *module\_clock* must be greater or equal to  $4 \text{ M} \times 16 = 64 \text{ MHz}$ .

### NOTE

The restriction that *peripheral\_clock* frequency must be higher or equal to 16x baud rate has been removed. There is no limitation on *peripheral\_clock* frequency to baud rate.

## 51.4.2.2 Maximum Baud Rate

The max baud rate the UART can support is determined by the max frequency of the *module\_clock* and logic synthesis results.

For example, if the SoC can provide the fastest *module\_clock* 66.5 MHz and the UART synthesis timing is acceptable under this constraint, the UART can transmit and receive serial data with the maximum baud rate  $66.5 \text{ M} / 16 = 4.15 \text{ Mbit/s}$ .

The UART supports serial IR interface low speed. In the low speed IrDA mode, the max baud rate is 115.2Kbit/s. To support the 115.2Kbit/s, *module\_clock* frequency must be higher or equal to 1.8432MHz.

## 51.4.2.3 Clocking in Low-Power Modes

The UART supports 2 low-power modes: DOZE and STOP.

In STOP mode (input pin *stop\_req* is at '1'), the UART doesn't need any clock. In this mode the UART can wake-up the ARM platform with the asynchronous interrupts (refer to [Low Power Modes](#)). An application of this feature is when the system must be waken-up by the arrival of a frame of characters.

- If before entering in Stop mode the software has enabled RTSDEN interrupt, then when RTS will change state (put at '0' by external device started to send), the

asynchronous interrupt will wake-up the system, *peripheral\_clock* and *module\_clock* will be provided to the UART before first start bit, so that no data will be lost.

- If RTS doesn't change state (already at '0' before entering in Stop mode), then wake-up interrupt (AWAKE) will be sent at the arrival of first Start bit (on falling edge). In this case, the UART must receive the *peripheral\_clock* and *module\_clock* during the first half of start bit to correctly receive this character (for example, at 115.2 Kbit/s, UART must receive *peripheral\_clock* and *module\_clock* at maximum 4.3 microseconds after falling edge of Start bit). If the UART receives *peripheral\_clock* and *module\_clock* too late, first character will be lost, and so should be dropped. Also, if autobaud detection is enabled, the first character won't be correctly received and another autobaud detection will need to be initiated.

In Doze mode, UART behavior is programmable through DOZE bit (UCR1[1]). If DOZE bit is set to '1', then UART is disabled in Doze mode, and in consequence, UART clocks can be switched-off (after being sure UART is not transmitting nor receiving). On the contrary, if DOZE bit is set to '0', UART is enabled and it must receive *peripheral\_clock* and *module\_clock*.

### 51.4.3 General UART Definitions

Definitions of terms that occurs the following discussions are given in this section.

- Bit Time-The period of time required to serially transmit or receive 1 bit of data (1 cycle of the baud rate frequency).
- Start bit-The bit time of a logic 0 that indicates the beginning of a data frame. A start bit begins with a 1-to-0 transition, and is preceded by at least 1 bit time of logic 1.
- Stop bit-1 bit time of logic 1 that indicates the end of a data frame.
- BREAK-A frame in which all of the data bits, including the stop bit, are logic 0. This type of frame is usually sent to signal the end of a message or the beginning of a new message.
- Mark - When no data is being sent, the serial port's transmit pin's voltage is 1 and is said to be in a MARK state.
- Space - The serial port can also be forced to keep the transmit pin at a 0 and is said to be the SPACE or BREAK state.
- Frame-A start bit followed by a specified number of data or information bits and terminated by a stop bit. The number of data or information bits depends on the format specified and must be the same for the transmitting device and the receiving device. The most common frame format is 1 start bit followed by 8 data bits (least significant bit first) and terminated by 1 stop bit. An additional stop bit and a parity bit also can be included.

- **Framing Error**-An error condition that occurs when the stop bit of a received frame is missing, usually when the frame boundaries in the received bit stream are not synchronized with the receiver bit counter. Framing errors can go undetected if a data bit in the expected stop bit time happens to be a logic 1. A framing error is always present on the receiver side when the transmitter is sending BREAKs. However, when the UART is programmed to expect 2 stop bits and only the first stop bit is received, this is not a framing error by definition.
- **Parity Error**-An error condition that occurs when the calculated parity of the received data bits in a frame does not match the parity bit received on the RX\_DATA input. Parity error is calculated only after an entire frame is received.
- **Idle**-One in NRZ encoding format and selectable polarity in IrDA mode.
- **Overrun Error**-An error condition that occurs when the latest character received is ignored to prevent overwriting a character already present in the UART receive buffer (RxFIFO). An overrun error indicates that the software reading the buffer (RxFIFO) is not keeping up with the actual reception of characters on the RX\_DATA input.

#### 51.4.3.1 RTS\_B - UART Request To Send

The UART Request To Send input controls the transmitter. The modem or other terminal equipment signals the UART when it is ready to receive by setting '0' on the RTS\_B pin.

Normally, the transmitter waits until this signal is active (low) before transmitting a character, however when the Ignore RTS (IRTS) bit is set, the transmitter sends a character as soon as it is ready to transmit. An interrupt (RTSD) can be posted on any transition of this pin and can wake the ARM platform from STOP mode on its assertion. When RTS\_B is set to '1' during a transmission, the UART transmitter finishes transmitting the current character and shuts off. The contents of the TxFIFO (characters to be transmitted) remain undisturbed. The operation of this input is the same regardless of whether the UART is in DTE or DCE mode.

#### 51.4.3.2 RTS Edge Triggered Interrupt

The input to the RTS\_B pin can be programmed to generate an interrupt on a selectable edge.

See the table below for summary of the operation of the RTS edge triggered interrupt (RTSF).

To enable the RTS\_B pin to generate an interrupt, set the request to send interrupt enable (RTSEN) bit (UCR2[4]) to 1. Writing 1 to the RTS\_B edge triggered interrupt flag (RTSF) bit (USR2[4]) clears the interrupt flag. The interrupt can occur on the rising edge, falling edge, or either edge of the RTS\_B input. The request to send edge control (RTEC) field (UCR2[10:9]) programs the edge that generates the interrupt. When RTEC is set to 0x00 and RTSEN = 1, the interrupt occurs on the rising edge (default). When RTEC is set to 0x01 and RTSEN = 1, the interrupt occurs on the falling edge. When RTEC is set to 0x1X and RTSEN = 1, the interrupt occurs on either edge. This is a synchronous interrupt. The RTSF bit is cleared by writing 1 to it. Writing 0 to RTSF has no effect.

**Table 51-8. RTS\_B Edge Triggered Interrupt Truth Table**

RTS_B	RTSEN	RTEC [1]	RTEC [0]	RTSF	Interrupt Occurs On...	interrupt_uart
X	0	X	X	0	Interrupt disabled	1
1->0	1	0	0	0	Rising edge	1
0->1	1	0	0	1	Rising edge	0
1->0	1	0	1	1	Falling edge	0
0->1	1	0	1	0	Falling edge	1
1->0	1	1	X	1	Either edge	0
0->1	1	1	X	1	Either edge	0

There is another RTS\_B interrupt that is not programmable. The status bit RTSD asserts the *interrupt\_uart* interrupt when the RTS\_B delta interrupt enable = 1. This is an asynchronous interrupt. The RTSD bit is cleared by writing 1 to it. Writing 0 to the RTSD bit has no effect.

### 51.4.3.3 DTR\_B - Data Terminal Ready

This signal indicates the general readiness of the Data Terminal Equipment (DTE). This signal is an input in DCE mode and an output in DTE mode. If the connection between the DCE and the DTE is established once, the DTR\_B signal must remain active throughout the whole connection time.

In general the DTR\_B and DSR\_B signals are responsible for establishing the connection. RTS\_B and CTS\_B are responsible for the data transfer and the transfer direction in the case of a half-duplex configuration. The DTR\_B signal is like a "main switch". If the DTR\_B signal is inactive the RTS\_B and CTS\_B signals have no effect. In DCE mode, an interrupt (DTRD) can be posted on any transition of this pin and can wake the ARM platform from STOP mode on its assertion.

### 51.4.3.4 DSR\_B - Data Set Ready

This signal indicates the general readiness of the DCE. This signal is an output in DCE mode and an input in DTE mode. The DCE uses this signal to inform the DTE that it is switched on, has completed all preparations and can communicate with the DTE.

In DTE mode, an interrupt (DTRD) can be posted on any transition of this pin and can wake the ARM platform from STOP mode on its assertion.

### 51.4.3.5 DTR\_B/DSR\_B Edge Triggered Interrupt

The DTR\_B input pin (DCE mode) or DSR\_B input pin (DTE mode) can be configured to cause an interrupt on a selectable edge.

See the table below for summary of the operation of the DTR/DSR edge triggered interrupt. To enable the interrupt, set the DTREN bit (UCR3[13]) to '1'. Write a "one" to the DTRF bit (USR2[13]) to clear the interrupt flag.

The interrupt can be configured to occur on either the rising, falling, or either edge of the DTR\_B/DSR\_B input. Write to the DPEC[1:0] bits (UCR3[15:14]) to program which edge will cause an interrupt. If the bits are set to 00b and DTREN = 1, the interrupt will occur on the rising edge (default). If the bits are set to 01b and DTREN = 1, the interrupt will occur on the falling edge. If the bits are set to 1Xb and DTREN = 1, the interrupt will occur on either edge.

**Table 51-9. DTR/DSR\_B Edge Triggered Interrupt Truth Table**

DTR_B / DSR_B	DTREN	DPEC[1]	DPEC[0]	DTRF	Interrupt occurs on:	interrupt_uart
X	0	X	X	0	turned off	1
1->0	1	0	0	0	rising edge	1
0->1	1	0	0	1	rising edge	0
1->0	1	0	1	1	falling edge	0
0->1	1	0	1	0	falling edge	1
1->0	1	1	X	1	either edge	0
0->1	1	1	X	1	either edge	0

### 51.4.3.6 DCD\_B - Data Carrier Detect

This signal is an output in DCE mode and an input in DTE mode. If used, the DCE device uses this signal to inform the DTE it has detected the carrier signal and the connection will be set up. This signal remains active while the connection remains established.

In DTE mode this input can trigger an interrupt on changing state. This is achieved by setting to '1' the interrupt enable bit (DCD, UCR3[9]). The change state is reflected in DCDDFLT (USR2[6]). Also, the state of the Data Carrier Detect input is mirrored in the status register DCDIN (USR2[5]).

### 51.4.3.7 RI\_B - Ring Indicator

This signal is an output in DCE mode and an input in DTE mode. If used, the DCE device uses this signal to inform the DTE that a ring just occurred.

In DTE mode this input can trigger an interrupt on changing state. This is achieved by setting to '1' the interrupt enable bit (RI, UCR3[8]). The change state is reflected in RIDFLT (USR2[10]). Also, the state of the Ring Indicator input is mirrored in the status register RIIN (USR2[9]).

### 51.4.3.8 CTS\_B - Clear To Send

This output pin serves two purposes. Normally, the receiver indicates that it is ready to receive data by asserting this pin (low). When the CTS\_B trigger level is programmed to trigger at 32 characters received and the receiver detects the valid start bit of the 33 character, it de-asserts this pin. The operation of this output is the same regardless of whether the UART is in DTE or DCE mode.

### 51.4.3.9 Programmable CTS\_B Deassertion

The CTS\_B output can also be programmed to deassert when the Rx FIFO reaches a certain level. Setting the CTS trigger level (UCR4[15:10]) at any value less than 32 deasserts the CTS\_B pin on detection of the valid start bit of the N + 1 character (where N is the trigger level setting). However, the receiver continues to receive characters until the Rx FIFO is full.

### 51.4.3.10 TX\_DATA - UART Transmit

This is the transmitter serial output. When operating in RS-232/RS-485 mode, NRZ encoded data is transmitted, and the data can be inverted (controlled by INVT (UCR3[1])) before transmitted. When operating in infrared mode, a 3/16 bit-period pulse is output for each 0 bit transmitted, and no pulse is output for each 1 bit transmitted.

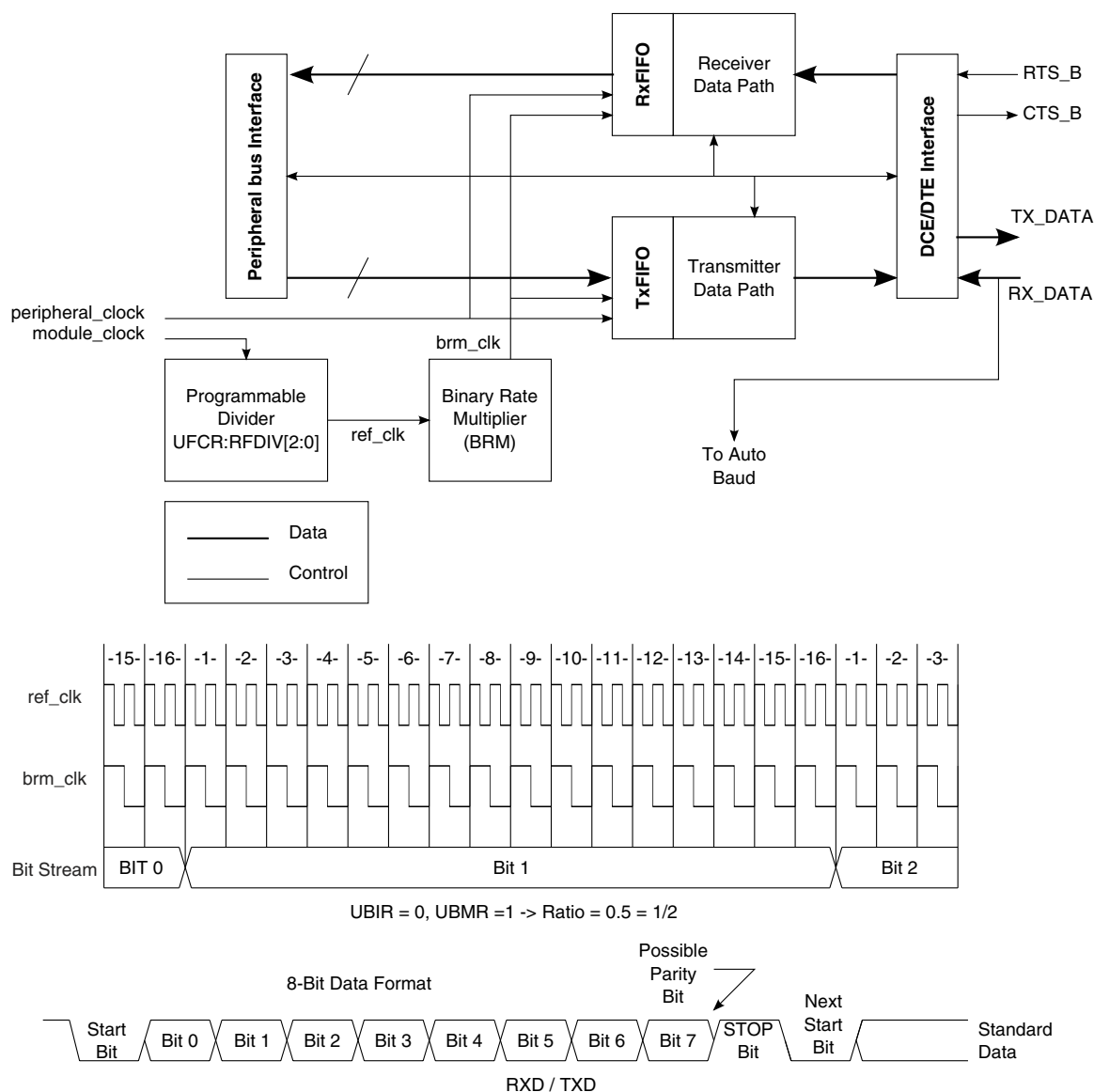
For RS-232/RS-485 applications, this pin must be connected to an RS-232/RS-485 transmitter. The operation of this output is the same regardless of whether the UART is in DTE or DCE mode. See [Figure 51-2](#).



### 51.4.3.11 RX\_DATA - UART Receive

This is the receiver serial input. When operating in RS-232/RS-485 mode, NRZ encoded data is expected, and the data can be inverted (controlled by INVR (UCR4[9])) before sampled. When operating in infrared mode, a narrow pulse is expected for each 0 bit received and no pulse is expected for each 1 bit received.

External circuitry must convert the IR signal to an electrical signal. RS-232/RS-485 applications require an external RS-232/RS-485 receiver to convert voltage levels. The operation of this input is the same regardless of whether the UART is in DTE or DCE mode. See the figure below.



**Figure 51-2. UART Simplified Block and Clock Generation Diagrams**

## 51.4.4 Transmitter

The transmitter accepts a parallel character from the ARM platform and transmits it serially. The start, stop, and parity (when enabled) bits are added to the character.

When the ignore RTS bit (IRTS) is set, the transmitter sends a character as soon as it is ready to transmit. RTS\_B can be used to provide flow-control of the serial data. When RTS\_B is set to '1', the transmitter finishes sending the character in progress (if any), stops, and waits for RTS\_B to be set to '0' again. Generation of BREAK characters and parity errors (for debugging purposes) is supported. The transmitter operates from the clock provided by the Binary Rate Multiplier(BRM). Normal NRZ encoded data is transmitted when the IR interface is disabled.

The transmitter FIFO (TxFIFO) contains 32 bytes. The data is written to TxFIFO by writing to the UTXD register with the byte data to the [7:0] bits. The data is written consecutively if the TxFIFO is not full. It is read (internally) consecutively if the TxFIFO is not empty. TXFULL bit (UTS[4]) can be used to control whether TxFIFO is full or not. The TxFIFO can be written regardless of the transmitter is disabled or enabled. If the UART is disabled, user can still write data into the TxFIFO correctly. But in this case the write access will yield to a transfer error.

### 51.4.4.1 Transmitter FIFO Empty Interrupt Suppression

The transmitter FIFO empty interrupt suppression logic suppresses the TXFE interrupt between writes to the TxFIFO.

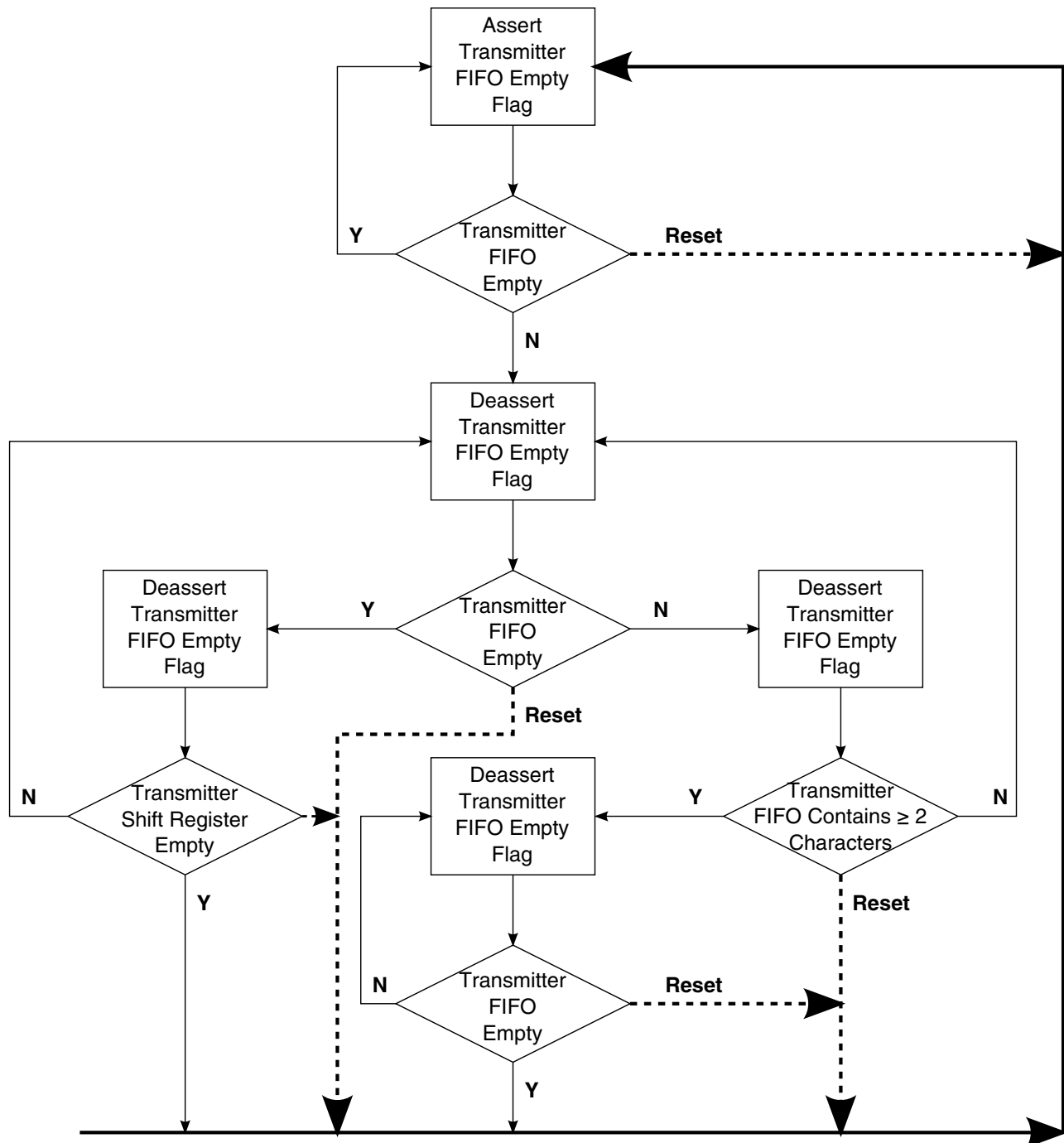
When TxFIFO is empty, the software can either send one or several characters. If the software sends one character, it would write the character into the UTXD register, then that character is immediately transferred to the transmitter shift register, assuming the transmitter is already enabled. Without interrupt suppression logic, the TXFE interrupt flag would be set immediately. But, with this logic, the interrupt flag is set when the last bit of the character has been transmitted, for example, before the transmission of the parity bit (if exists) and the stop bit(s).

So, the suppression logic doesn't immediately send the TXFE interrupt flag. It allows the software to write another character to the TxFIFO before the interrupt flag is asserted.

When the transmitter shift register empties before another character is written to the TxFIFO, the interrupt flag is asserted. Writing data to the TxFIFO would release the interrupt flag. The interrupt flag is asserted on the following conditions:

- System Reset

- UART software reset
- When a single character has been written to Transmitter FIFO and then the Transmitter FIFO and the Transmitter Shift Register become empty until another character is written to the Transmitter FIFO
- The last character in the TxFIFO is transferred to the shift register, when TxFIFO contains two or more characters. See the figure below.



### Figure 51-3. Transmitter FIFO Empty Interrupt Suppression Flow Chart

### 51.4.4.2 Transmitting a Break Condition

Asserting SNDBRK bit of the UCR1 Register forces the transmitter to send a break character (continuous zeros). The transmitter will finish sending the character in progress (if any) before sending break until this bit is reset.

The user is responsible to ensure that this bit is high for long enough to generate a valid BREAK. The transmitter samples SNDBRK after every bit is transmitted. Following completion of the BREAK transmission, the UART will transmit two mark bits. The user can continue to fill the FIFO and any character remaining will be transmitted when the break is terminated.

### 51.4.5 Receiver

See the figure below for the receiver flow chart.

The receiver accepts a serial data stream and converts it into parallel characters. When enabled, it searches for a start bit, qualifies it, and samples the following data bits at the bit-center.

Jitter tolerance and noise immunity are provided by sampling at a 16x rate and using voting techniques to clean up the samples. Once the start bit is found, the data bits, parity bit (if enabled), and stop bits (either 1 or 2 depending on user selection) are shifted in. Parity is checked and its status reported in the URXD register when parity is enabled. Frame errors and BREAKs are also checked and reported. When a new character is ready to be read by the ARM platform from the Rx FIFO, the receive data ready (RDR = USR2[0]) bit is asserted and an interrupt is posted (if DREN = UCR4[0] = 1). If the receiver trigger level is set to 2 (RXTL[5:0] = UFCR[5:0] = 2), and 2 chars have been received into Rx FIFO, the receiver ready interrupt flag (RRDY = USR1[9]) is asserted and an interrupt is posted if the receiver ready interrupt enable bit is set (RRDYEN = UCR1[9] = 1). If the UART Receiver Register (URXD) is read once, and in consequence there is only 1 character in the Rx FIFO, the interrupt generated by the RDR bit is automatically cleared. The RRDY bit is cleared when the data in the Rx FIFO falls below the programmed trigger level.

Normal NRZ encoded data is expected when the IR interface is disabled. The Rx FIFO contains 32 half-word entries. Characters received are written consecutively into this FIFO. If the FIFO is full and a 33rd character is received, this character will be ignored and the USR2[ORE] bit will be set.

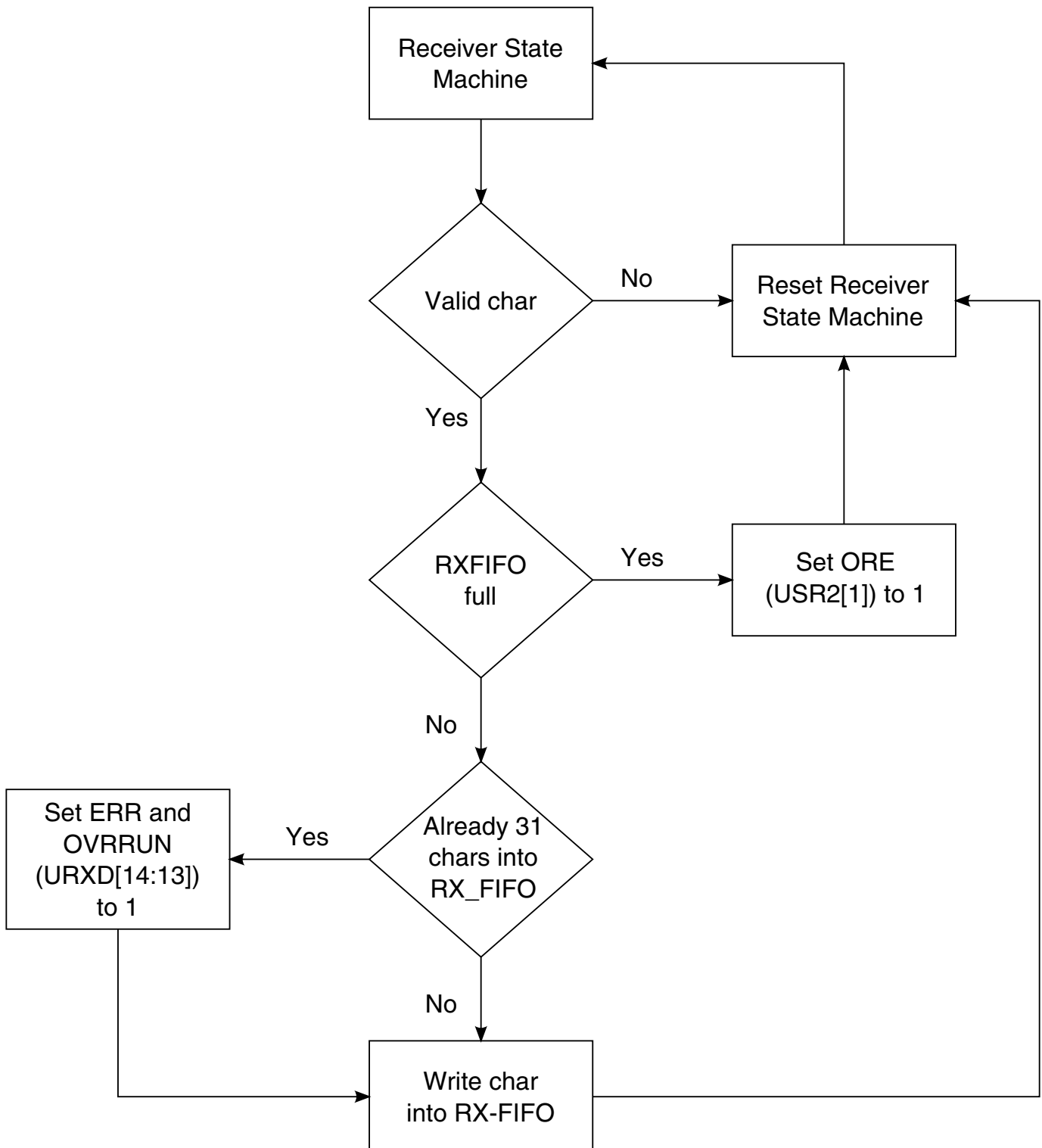


Figure 51-4. Receiver Flow Chart

### 51.4.5.1 Idle Line Detect

The receiver logic block includes the ability to detect an idle line. Idle lines indicate the end or the beginning of a message.

For an idle condition to occur:

- RxFIFO must be empty and
- RX\_DATA pin must be idle for more than a configured number of frames (ICD[1:0] = UCR1[11:10]).

When the idle condition detected interrupt enable (IDEN = UCR1[12]) is set and the line is idle for 4 (default), 8, 16, or 32 (maximum) frames, the detection of an idle condition flags an interrupt (see the table below). When an idle condition is detected, the IDLE (USR2[12]) bit is set. Clear the IDLE bit by writing 1 to it. Writing 0 to the IDLE bit has no effect.

**Table 51-10. Detection Truth Table**

IDEN	ICD [1]	ICD [0]	IDLE	<i>interrupt_uart</i>
0	X	X	0	1
1	0	0	asserted after 4 idle frames	asserted after 4 idle frames
1	0	1	asserted after 8 idle frames	asserted after 8 idle frames
1	1	0	asserted after 16 idle frames	asserted after 16 idle frames
1	1	1	asserted after 32 idle frames	asserted after 32 idle frames
<b>NOTE:</b> This table assumes that no other interrupt is set at the same time this interrupt is set for the <i>interrupt_uart</i> signal. This table shows how this interrupt affects the <i>interrupt_uart</i> signal.				

During a normal message there is no idle time between frames. When all of the information bits in a frame are logic 1s, the start bit ensures that at least one logic 0 bit time occurs for each frame so that the IDLE bit is not asserted.

### 51.4.5.2 Aging Character Detect

The receiver block also includes the possibility to detect when at least one character has been sitting into the RxFIFO for a time corresponding to 8 characters. This aging character capability allows the UART to inform the ARM platform that there is less character into the RxFIFO than the Rx trigger and, no new character has been detected on the RXD line.

The aging capability is a timer which starts to count as soon as the RxFIFO is not empty and its trigger level is not reached (RRDY=0). This counter is reset when either a RxFIFO read is performed or another character starts to present on the RXD line. If none of those two events occurs, the bit AGTIM (USR1[8]) is set when the counter has

measured a time corresponding to 8 characters. AGTIM is cleared by writing a 1 to it. AGTIM can flag an interrupt to ARM platform on *interrupt\_uart* if ATEN (UCR2[3]) has been set.

To summarize, AGTIM is set when:

- There is at least one character into RxFIFO.
- No read has occurred on RxFIFO and RXD line has stayed high, for a time corresponding to 8 characters.
- The RxFIFO trigger is not reached (RRDY=0)

### 51.4.5.3 Receiver Wake

The WAKE bit (USR2[7]) is set when the receiver detects a qualified Start bit. For this, two conditions must be fulfilled, firstly a falling edge on RX\_DATA line must be detected and secondly the RX\_DATA line must stay at low level for more than a half-bit duration.

When the wake interrupt enable WKEN (UCR4[7]) bit is enabled, the receiver flags an interrupt ( *interrupt\_uart*) if the WAKE status bit is set. The WAKE bit is cleared by writing 1 to it. Writing 0 to the WAKE bit has no effect. The WAKE status bit can be asserted in either serial RS-232 mode or IR mode. The generation of the WAKE interrupt needs the clock *module\_clock* .

When the asynchronous wake interrupt (AWAKE) is enabled (AWAKEN = UCR3[4] = 1), and the ARM platform is in STOP mode, and UART clocks have been shut-off, then a falling edge detected on the receive pin (RX\_DATA) asserts the AWAKE bit (USR1[4]) and the *interrupt\_uart* interrupt to wake the ARM platform from STOP mode. Re-enable UART clocks and clear the AWAKE bit by writing 1 to it. Writing 0 to the AWAKE bit has no effect. When IR interface is enabled (UCR1[7]=1), the AWAKE bit is always not asserted. The generation of the asynchronous AWAKE interrupt does not need any clocks.

In IR mode, if the asynchronous IR WAKE interrupt is enabled (AIRINTEN = UCR3[5] = 1), and if the ARM platform is in STOP mode (UART clocks are off when ARM platform in STOP mode), then the detection of a falling edge on the receive pin (RXD\_IR), asserts the AIRINT bit (USR1[5]), and the *interrupt\_uart* interrupt. This interrupt wakes the ARM platform from STOP mode. Software re-enables UART clocks and clear the AIRINT bit by writing 1 to it. Writing 0 to the AIRINT bit has no effect. When IR interface is disabled (UCR1[7]=0), the AIRINT bit is always not asserted. The generation of the asynchronous AIRINT interrupt does not need any clocks.

Recommended procedure for programming the asynchronous interrupts is to first clear them by writing 1 to the appropriate bit in the UART Status Register 1 (USR1). Poll or enable the interrupt for the Receiver IDLE Interrupt Flag (RXDS) in the USR1. When asserted, the RXDS bit indicates to the software that the receiver state machine is in the idle state, the next state is idle, and the RX\_DATA pin is idle (high). After following this procedure, enable the asynchronous interrupt and enter STOP mode.

#### 51.4.5.4 Receiving a BREAK Condition

A BREAK condition is received when the receiver detects all 0s (including a 0 during the bit time of the stop bit) in a frame. The BREAK condition asserts the BRCD bit (USR2[2]) and writes only the first BREAK character to the RxFIFO. Clear the BRCD bit by writing 1 to it. Writing 0 to the BRCD bit has no effect.

Asserting BRCD would generate an interrupt on *interrupt\_uart*. The interrupt generation can be masked using the control bit BKEN (UCR4[2]). Receiving a break condition will also effect the following bits in the receiver register URXD:

URXD(11) = BRK. While high this bit indicates that the current char was detected as a break.

URXD(12) = FRMERR. The frame error bit will always be set when BRK is set.

URXD(10) = PRERR. If odd parity was selected the parity error bit will also be set when BRK is set.

URXD(14) = ERR. The error detect bit indicates that the character present in the rx data field has an error status. This can be asserted by a break.

#### 51.4.5.5 Vote Logic

The vote logic block provides jitter tolerance and noise immunity by sampling with respect to a 16x clock (*brm\_clk*) and using voting techniques to clean up the samples. The voting is implemented by sampling the incoming signal constantly on the rising edge of the *brm\_clk*.

See [Figure 51-5](#). The receiver is provided with the majority vote value, which is 2 out of the 3 samples. For examples of the majority vote results of the vote logic, see the following table.

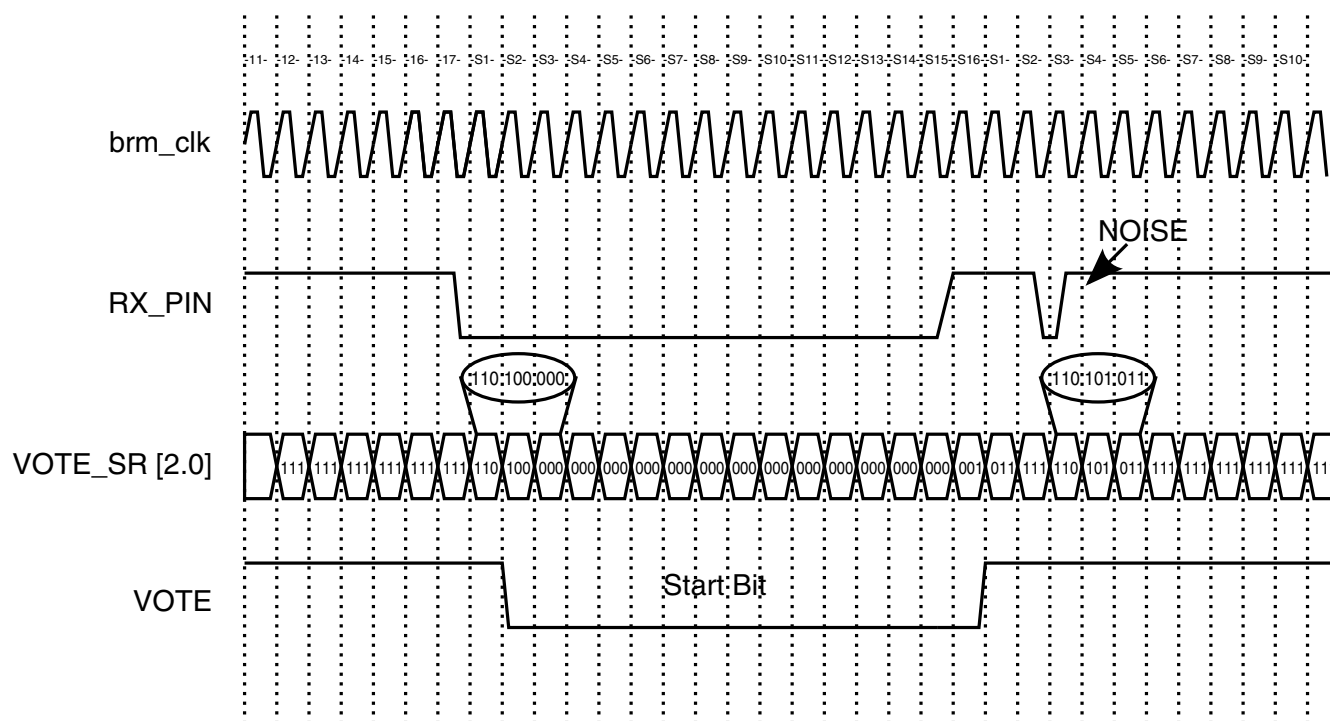


**Table 51-11. Majority Vote Results**

Samples	Vote
000	0
101	1
001	0
111	1

The vote logic captures a sample on every rising edge of *brm\_clk*, however the receiver uses 16x oversampling to take its value in the middle of the sample character.

The receiver starts to count when the Start bit is set however it does not capture the contents of the RxFIFO at the time the Start bit is set. The start bit is validated when 0s are received for 7 consecutive 1/16 of bit times following the 1-to-0 transition. Once the counter reaches 0xF, it starts counting on the next bit and captures it in the middle of the sampling frame (see [Table 51-11](#)). All data bits are captured in the same manner. Once the stop bit is detected, the receiver shift register (SIPO\_OUT) data is parallel shifted to the RxFIFO.

**Figure 51-5. Majority Vote Results**

A new feature has been recently implemented, it allows to re-synchronize the counter on each edge of RX\_DATA line. This is automatic and allows to improve the immunity of UART against signal distortion.

There is a special case when the *brm\_clk* frequency is too low and is unable to capture a 0 pulse in IrDA. In this case, the software must set the IRSC (UCR4[5]) bit so that the reference clock (after internal divider) is used for the voting logic. The pulse is validated by counting the length of the pulse.

Refer to [Infrared Interface](#) for more details.

### 51.4.5.6 Baud Rate Automatic Detection Logic

When the baud rate automatic detection logic is enabled, the UART locks onto the incoming baud rate. To enable this feature, set the automatic detection of baud rate bit (ADBR = UCR1[14] = 1) and write 1 to the ADET bit (USR2[15]) to clear it.

When ADET=0 and ADBR =1, the detection starts. Then, once the beginning of start bit (transition from 1-to-0 of RX\_DATA) has been detected, UART starts a counter (UBRC) working at reference frequency. Once the end of start bit is detected (transition from 0-to-1 of RX\_DATA), the value of UBRC - 1 is directly copied into UBMR register. UBIR register is filled with 0x000F.

So, at the end of start bit, registers gets following values:

```
UBRC = number of reference clock periods (after divider) during Start bit.
UBIR = 0x000F
UBMR = UBRC - 1
```

The updated values of the 3 registers can be read.

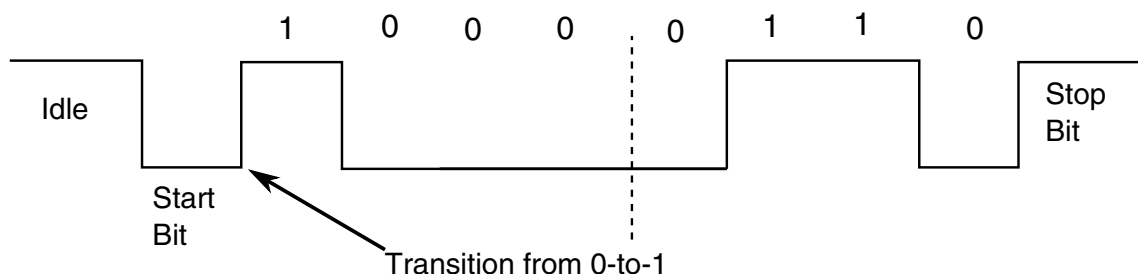
See [Table 51-12](#) for list of parameters for baud rate detection and [Figure 51-6](#) for baud rate detection protocol diagram.

If any of the UART BRM registers are simultaneously written by the baud rate automatic detection logic and by the peripheral data bus, the peripheral data bus would have lower priority.

**Table 51-12. Baud Rate Automatic Detection**

ADBR	ADET	Baud Rate Detection	<i>interrupt_uart</i>
0	X	Manual Configuration	1
1	0	Auto Detection Started	1
1	1	Auto Detection Complete	0

**NOTE:** This table assumes that no other interrupt is set at the same time this interrupt is set for the *interrupt\_uart* signal.



**Note:** LSB Transmitted first.

**Figure 51-6. Baud Rate Detection Protocol Diagram**

#### 51.4.5.6.1 Baud Rate Automatic Detection Protocol

The receiver must receive an ASCII character "A" or "a" to verify proper detection of the incoming baud rate. When an ASCII character "A" (0x41) or "a" (0x61) is received and no error occurs, the Automatic Detect baud rate bit is set (ADET=1) and if the interrupt is enabled (ADEN=UCR1[15]=1), an interrupt *interrupt\_uart* is generated.

When an ASCII character "A" or "a" is not received (because of a bit error or the reception of another character), the auto detection sequence restarts and waits for another 1-to-0 transition.

As long as ADET = 0 and ADBR = 1, the UART continues to try to lock onto the incoming baud rate. Once the ASCII character "A" or "a" is detected and the ADET bit is set, the receiver ignores the ADBR bit and continues normal operation with the calculated baud rate.

The UART interrupt is active ( *interrupt\_uart* = 0) as long as ADET = 1 and ADBR = 1. This can be disabled by clearing the automatic baud rate detection interrupt enable bit (ADEN = 0). Before starting an automatic baud rate detection sequence, set ADET = 0 and ADBR = 1.

The RxFIFO must contain the ASCII character "A" or "a" following the automatic baud rate detection interrupt.

The 16-bit UART Baud Rate Count Register (UBRC) is reset to 4 and stays at 0xFFFF when an overflow occurs. The UBRC register counts (measures) the duration of start bit. When the start bit is detected and counted, the UART Baud Rate Count Register retains its value until the next automatic baud rate detection sequence is initiated.

The Baud Rate Count Register counts only when auto detection is enabled.

### 51.4.5.6.2 New Baud Rate Determination

In order to fight against the problems caused by the distortion and the noise on the RX\_DATA line, the duration of the baud rate measurement has been extended.

Previously, as described above, this determination was based on the measurement of the START bit duration. Now, this measurement is based on the duration of START bit + bit0. Bit0 is the first bit following the START bit. In fact, the counter which is started at the falling edge of START bit is no longer stopped at next rising edge (end of START bit), but it is stopped at the next falling edge (end of bit0). As the character sent is always a "A" (41h) or a "a" (61h), this second falling edge will always be present and it will indicate the end of bit0. Once this counter is stopped, the result is divided by 2 and used by the BRM to determine the incoming baud rate.

#### NOTE

UBRC register contains the result of this division by two, in consequence it reflects the measurement of the duration of one bit.

#### 51.4.5.6.2.1 New Autobaud Counter Stopped bit and Interrupt

A new bit has been added in USR2 register: ACST (USR2[11]). This bit is set immediately after the determination of the baud rate.

So,

- if ADNIMP is not set (default), ACST is set to 1 after the end of bit0,
- If ADNIMP is set to 1, ACST is set to 1 at the end of START bit.

If ACIEN (UCR3[0]) is set to 1, ACST will flag an interrupt on *interrupt\_uart* signal. This interrupt informs the ARM platform that the BRM has just been set with the result of the bit length measurement. If needed, the ARM platform can perform a read of UBMR (or UBRC) register and determine by itself the baud rate measured. Then the ARM platform has the possibility to correct the BRM registers with the nearest standardized baud rate.

#### NOTE

ACST is set only if ADBR is set to 1, for example, the UART is autobauding.

Clear the ACST bit by writing 1 to it. Writing 0 to the ACST bit has no effect.

## 51.4.6 Escape Sequence Detection

An escape sequence typically consists of 3 characters entered in rapid succession (such as +++). Because these are valid characters by themselves, the time between characters determines if it is a valid escape sequence.

Too much time between two of the "+" characters is interpreted as two "+" characters, and not part of an escape sequence.

The software chooses the escape character and writes its value to the UART Escape Character Register (UESC). The software must also enable escape detection feature by setting ESCEN (UCR2[11]) to 1. The hardware compares this value to incoming characters in the RxFIFO. When an escape character is detected, the internal escape timer starts to count. The software specifies a time-out value for the maximum allowable time between 2 successive escape characters (see the table below). The escape timer is programmable in intervals of 2 ms to a maximum interval of 8.192 seconds.

**Table 51-13. Escape Timer Scaling**

UTIM Register	Maximum Time Between Specified Escape Characters
0x000	2 ms
0x001	4 ms
0x002	6 ms
0x003	8 ms
0x004	10 ms
...	...
0F8	498 ms
0F9	500 ms
...	...
9C3	5 s
...	...
FFD	8.188 s
FFE	8.190 s
FFF	8.192 s
<b>NOTE:</b> To calculate the time interval: $(\text{UTIM\_Value} + 1) \times 0.002 = \text{Time\_Interval}$ Example: $(09C3 + 1) \times 0.002 = 5 \text{ s.}$	

The escape sequence detection feature is available for all the reference frequencies. Before using Escape Sequence Detection, the user must fill the ONEMS register. This 24-bit register must contain the value of the UART internal frequency divided by 1000. The internal frequency is obtained after the UART internal divider which is applied on *module\_clock* clock.

Example I:

- If the input clock *module\_clock* frequency is 66.5 MHz.
- And if the input clock *module\_clock* is divided by 2 with the internal divider:  
UFCR[9:7] = 3'b100

$$\text{ONEMS} = \frac{66.5 \times 10^6}{2 \times 1000} = 33250 = 81\text{E2h}$$

**Figure 51-7. Calculation of Frequency for ONEMS Register**

Example II:

- If the input clock *module\_clock* frequency is 66.5 MHz.
- And if the input clock *module\_clock* is divided by 1 with the internal divider:  
UFCR[9:7] = 3'b101

$$\text{ONEMS} = \frac{66.5 \times 10^6}{1000} = 66500 = 103\text{C4h}$$

**Figure 51-8. Calculation of Frequency for ONEMS Register**

The escape sequence detection interrupt is asserted when the escape sequence interrupt enable (ESCI) bit is set and an escape sequence is detected (ESCF set). Clear the ESCF bit by writing 1 to it. Writing 0 to the ESCF bit has no effect.

## 51.5 Binary Rate Multiplier (BRM)

The BRM sub-block receives *ref\_clk* (*module\_clock* clock after divider). From this clock, and with integer and non-integer division, BRM generates a 16x baud rate clock .

The UART transmitter will shift data out based on this 16x baud rate clock. The UART receiver will sample the serial data line based on this 16x baud rate clock. The input and output frequency ratio is programmed in the UART BRM Incremental Register (UBIR) and UART BRM MOD Register (UBMR). The output frequency is divided by the input frequency to produce this ratio. For integer division, set the UBIR = 0x000F and write the divisor to the UBMR register. All values written to these registers must be one less than the actual value to eliminate division by 0 (undefined), and to increase the maximum range of the registers.

Updating the BRM registers requires writing to both registers. The UBIR register must be written before writing to the UBMR register. If only one register is written to by the software, the BRM continues to use the previous values.

The following examples show how to determine what values are to be programmed into UBIR and UBMR for a given reference frequency and desired baud rate. The following equation can be used to help determine these values:

$$\text{BaudRate} = \frac{\text{Ref Freq}}{\left( 16 \times \frac{\text{UBMR} + 1}{\text{UBIR} + 1} \right)}$$

**Figure 51-9. Frequency and Baud Rate for UBIR and UBMR**

With:

Reference Frequency (Hz): UART Reference Frequency (*module\_clock* after RFDIV divider)

Baud Rate (bit/s): Desired baud rate.

Integer Division ÷ 21

Reference Frequency = 19.44 MHz

UBIR = 0x000F

UBMR = 0x0014

Baud Rate = 925.7 kbit/s

### NOTE

Observe that each value written to the registers is one less than the actual value.

Non-Integer Division

## Infrared Interface

Reference Frequency = 16 MHz  
Desired Baud Rate = 920 Kbits/s

$$\frac{UBMR + 1}{UBIR + 1} = \frac{RefFreq}{16 \times BaudRate} = \frac{16 \times 10^6}{16 \times 920 \times 10^3} = 1.087$$

Ratio = 1.087 = 1087 / 1000  
UBIR = 999 (decimal) = 0x3E7  
UBMR = 1086 (decimal) = 0x43E  
Non-Integer Division  
Reference Frequency = 25 MHz  
Desired Baud Rate = 920 kbit/s  
Ratio = 1.69837 = 625 / 368  
UBIR = 367 (decimal) = 0x16F  
UBMR = 624 (decimal) = 0x270

### Non-Integer Division

Reference Frequency: 30 MHz  
Desired Baud Rate = 115.2 kbit/s  
Ratio = 16.276043 = 65153 / 4003  
UBIR = 4002 (decimal) = 0x0FA2  
UBMR = 65152 (decimal) = 0xFE80

## 51.6 Infrared Interface

### 51.6.1 Generalities-Infrared

The Infrared interface is selected when IREN (UCR1[7]) is set to 1.

The Infrared Interface is compatible with IrDA Serial Infrared Physical Layer Specification. In this specification, a "zero" is represented by a positive pulse, and a "one" is represented by no pulse (line remains low).

In the UART:

In TX: For each "zero" to be transmitted, a narrow positive pulse which is 3/16 of a bit time is generated. For each "one" to be transmitted no pulse is generated (output is low). External circuitry has to be provided to drive an Infrared LED.

In RX: When receiving, a narrow negative pulse is expected for each "zero" transmitted while no pulse is expected for each "one" transmitted (input is high).

#### NOTE

Rx part of IR block expects to receive an inverted signal compared to IrDA specification. Circuitry external to the IC transforms the Infrared signal to an electrical signal.



The IR interface has an edge triggered interrupt (IRINT). This interrupt validates a zero bit being received. This interrupt is enabled by writing a "one" to ENIRI bit.

The behavior of Infrared Interface is determined by 3 bits INVT (UCR3[1]), INVR (UCR4[9]) and IRSC (UCR4[5]).

### 51.6.2 Inverted Transmission and Reception bits (INVT & INVR)

The values of INVT and INVR depend of the IrDA transceiver connected on the TXD\_IR and RXD\_IR pins of the UART. If this transceiver is not inverting on both paths Tx and Rx, a Zero is represented by a positive pulse and a One is represented by no pulse (line remains low). In this case, the bit INVT must be set to 0 and the bit INVR must be set to 1 (because Rx IR block expects an inverted signal).

On the contrary user must set INVT=1 and INVR=0 if both paths of the transceiver are inverting, that is, a Zero is represented as a negative pulse and a One is represented by no pulse (line remains high). The transceiver can also be inverting on only one path (Tx or Rx), in this case INVT and INVR must be together equal to 1 or to 0, depending on which path is inverted.

### 51.6.3 InfraRed Special Case (IRSC) Bit

The value to apply to IRSC bit is based on 2 parameters: the baud rate and the Minimum Pulse Duration (MPD) of the transceiver.

According to IrDA Standard Specification, for SIR (Serial IR) baud rates from 2.4 Kbit/s to 115.2 Kbit/s this nominal pulse duration is equal to 3/16 of a bit duration (at the selected baud rate). But, for all the baud rates a Minimum Pulse Duration is also specified. According to IrDA Standard, a Zero is represented by a light pulse, so the IrDA transceiver can't emit a light pulse shorter than the MPD. For SIR, the MPD is constant and equal to 1.41  $\mu$ s.

But user must take into account the electrical MPD associated with the transceiver on the receiver path. Typically this value is 2.0  $\mu$ s, but for some manufacturers MPD can go down to 1.0  $\mu$ s.

In order to understand the meaning of IRSC bit, one must understand how the RX path works in IrDA mode.

When the UART is in IrDA mode, a Zero is not only detected by the state of the RXD\_IR line, but also with the duration of the pulse. This pulse duration can be measured with 2 different clocks. In this case, clock is selected with the IRSC bit.

- If IRSC = 0, the clock used is the BRM clock.
- If IRSC = 1, the clock used is the UART internal clock (UART clock after the divider (RFDIV)).

In normal operation, IRSC=0. This means that at any time, the user must ensure that the frequency of BRM\_clock is high enough to measure the pulse. The pulse must last at least 2 BRM clock cycles. If this condition is not fulfilled, IRSC must be set to 1.

Let's examine two examples, for a Minimum Pulse Duration equal to the MPD from the IrDA SIR specification (i.e., 1.41  $\mu$ s).

#### 1: Calculation of BRM Clock Period (Clock Period < 1.41 $\mu$ s)

The user wants to receive IrDA data at 115.2 Kbit/s. The UBIR and UBMR registers are set in order to create the BRM\_clock with a frequency of  $16 \times \text{baud rate} = 16 \times 115.2\text{K} = 1.843\text{ MHz}$ . But at the same time, in order to correctly detect the pulse, the user must be sure that  $2 \times \text{BRM\_clock period}$  is lower than 1.41  $\mu$ s. Let's check:

$$\text{BRM\_clock period} = 1/1843000 = 542\text{ ns}$$

So  $2 \times \text{BRM\_clock period} = 1.09\text{ }\mu\text{s} < 1.41\text{ }\mu\text{s}$ . It is fine.

#### 2: Calculation of BRM Clock Period (Clock Period > 1.41 $\mu$ s)

This time the user wants to receive at 19.2 Kbit/s. So, the BRM\_clock is set to  $16 \times 19200 = 307.2\text{ kHz}$ . Let's check if  $2 \times \text{BRM\_clock period} < 1.41\text{ }\mu\text{s}$ :

$$1. \text{ BRM\_clock period} = 1/307200 = 3.25\text{ }\mu\text{s}$$

So  $2 \times \text{BRM\_clock period} = 6.50\text{ }\mu\text{s} >> 1.41\text{ }\mu\text{s}$ . It doesn't work.

So, in this case, the BRM clock can't be used to measure the pulse duration and the user must select the UART internal clock by setting IRSC = 1.

### NOTE

Like for Escape character detection, when IR Special Case is enabled (IRSC=1), the UART must measure a duration. In order to do that, the user must fill the ONEMS register. Refer to [Escape Sequence Detection](#).

### 51.6.4 IrDA interrupt

Serial infrared mode (SIR) uses an edge triggered interrupt flag IRINT (USR2[8]). When INVR = 0, detection of a falling edge on the RXD pin asserts the IRINT bit. When INVR = 1, detection of a rising edge on the RXD pin asserts the IRINT bit. When IRINT and ENIRI bits are both asserted, the *interrupt\_uart* interrupt is asserted. Clear the IRINT bit by writing 1 to it. Writing 0 to the IRINT bit has no effect.

### 51.6.5 Conclusion about IrDA

Before using the UART in IrDA, the baud rate limit must be calculated. This baud rate limit will inform the user if IRSC bit has to be set or not.

Let's determine this limit:

As already described, if IRSC = 0, the following condition must always be fulfilled

$$2 \times \text{BRMClockPeriod} < \text{MinPulseDuration}$$

**Figure 51-10. Calculation of Baud Rate**

So,

$$\text{BRMClockFrequency} > \frac{2}{\text{MPD}}$$

So, knowing BRM\_clock frequency = 16 \* Baud Rate, we get:

$$\text{BaudRate} > \frac{1}{8 \times \text{MinPulseDuration}}$$

So, the user needs to set IRSC = 0 when:

- If Minimum Pulse Duration = 2.5 us and Baud Rate > 50 Kbit/s.
- If Minimum Pulse Duration = 2.0 us and Baud Rate > 62.5 Kbit/s.
- If Minimum Pulse Duration = 1.41 us and Baud Rate > 88.6 Kbit/s.

#### **NOTE**

For baud rates lower than the limit, IRSC must be set to 1.

## 51.6.6 Programming IrDA Interface

### 51.6.6.1 High Speed

As an example, the following sequence can be used to program the IrDA interface in order to send and receive characters at 115.2 Kbit/s.

Assumptions:

- Input UART clock = 90 MHz
- Internal clock divider = 3 (divide Input UART clock by 3)
- Baud rate = 115.2 Kbit/s
- IrDA transceiver is not inverting on both channels: for Tx and Rx, a Zero is represented by a positive pulse, and a One is represented by no pulse (line stays low).
- Interrupt: Sent to ARM platform when 1 char is received into the Rx FIFO (RDR)

Registers values and Programming orders:

```

UCR1 = 0x0085
UCR1[7] = IREN = 1: Enable IR interface
UCR1[0] = UARTEN = 1: Enable UART
UTS = 0x0000
UFCR = 0x0981
TXTL[5:0] = 0x02: Default value
RFDIV[2:0] = 0x3: Divide Input UART clock by 3 (resulting internal clock is 30 MHz)
RXTL[5:0] = 0x01: Default value
UBIR = 0x0202
UBMR = 0x20BE Baud rate = 115.2 kbit/s with internal clock = 30 MHz
UCR2 = 0x4027
UCR2[14] = IRTS = 1: Ignore level of RTS input signal
UCR2[5] = WS = 1: Characters are 8-bit length
UCR2[2] = TXEN = 1: Enable Rx path
UCR2[1] = RXEN = 1: Enable Tx path
UCR2[0] = SRST_B = 1: No software reset
UCR3 = 0x0000

UCR4 = 0x8201
CTSTL[5:0] = 0x20: Default value
UCR4[9] = INVR = 1: Inverted Infrared Reception (because IrDA transceiver is not inverting)
UCR4[1] = DREN = 1: To enable RDR interrupt (sent when one char is received)

```

The UART is ready to send a character as soon as there is a write into UTXD register. And an interrupt will be sent to ARM platform when a character is received.

### 51.6.6.2 Low Speed

This time, we keep the same assumptions but the speed is now 9.6 Kbit/s. So, this baud rate is below the limit (even with a Min. Pulse Duration of 2.5 us) and thus IRSC must be set to 1.

**Assumptions:**

- Input UART clock = 90 MHz
- Internal clock divider = 3 (divide Input UART clock by 3)
- Baud rate = 9.6 Kbit/s
- IrDA transceiver is not inverting on both channels: for Tx and Rx, a Zero is represented by a positive pulse, and a One is represented by no pulse (line stays low).
- Interrupt: Sent to ARM platform when 1 char is received into the Rx FIFO (RDR).

**Registers values and Programming orders:**

```

UCR1 = 0x0085
UCR1[7] = IREN = 1: Enable IR interface
UCR1[0] = UARTEN = 1: Enable UART
UFCR = 0x0981
UFCR[15:10] = TXTL[5:0] = 0x02: Default value
RFDIV[2:0] = 0x3: Divide Input UART clock by 3 (resulting internal clock is 30 MHz)
UFCR[5:0] = RXTL[5:0] = 0x01: Default value
UBIR = 0x00FF
UBMR = 0xC354 Baud rate = 9.6 kbit/s with internal clock = 30 MHz
UCR2 = 0x4027
UCR2[14] = IRTS = 1: Ignore level of RTS input signal
UCR2[5] = WS = 1: Characters are 8-bit length
UCR2[2] = TXEN = 1: Enable Rx path
UCR2[1] = RXEN = 1: Enable Tx path
UCR2[0] = SRST_B = 1: No software reset
UCR3 = 0x0000
UCR3[1] = INVT = 0: Positive pulse represents 0.
UCR4 = 0x8221
UCR4[15:10] = CTSTL[5:0] = 0x20: Default value
UCR4[9] = INVR = 1: Inverted Infrared Reception (because IrDA transceiver is not inverting)
UCR4[5] = IRSC = 1: Because data rate is below the limit and thus the UART internal clock is
used to measure the pulse duration.
UCR4[1] = DREN = 1: To enable RDR interrupt (sent when one char is received)

```

The UART is now ready to send a character as soon as there is a write into UTXD register. An interrupt will be sent to ARM platform when a character is received.

## 51.7 9-bit RS-485 Mode

### 51.7.1 Generalities

The UART provides a 9-bit mode to facilitate multidrop (RS-485) network communication. To enable this mode, set MDEN bit in the UMCR register to 1. When 9-bit RS-485 mode is enabled, UART transmitter can transmit the ninth bit (9<sup>th</sup> bit) set by TXB8, and UART receiver can differentiate between data frames (9<sup>th</sup> bit = 0) and address frames (9<sup>th</sup> bit = 1).

The CTS\_B pin can be used to control RS-485 output driver outside the chip.

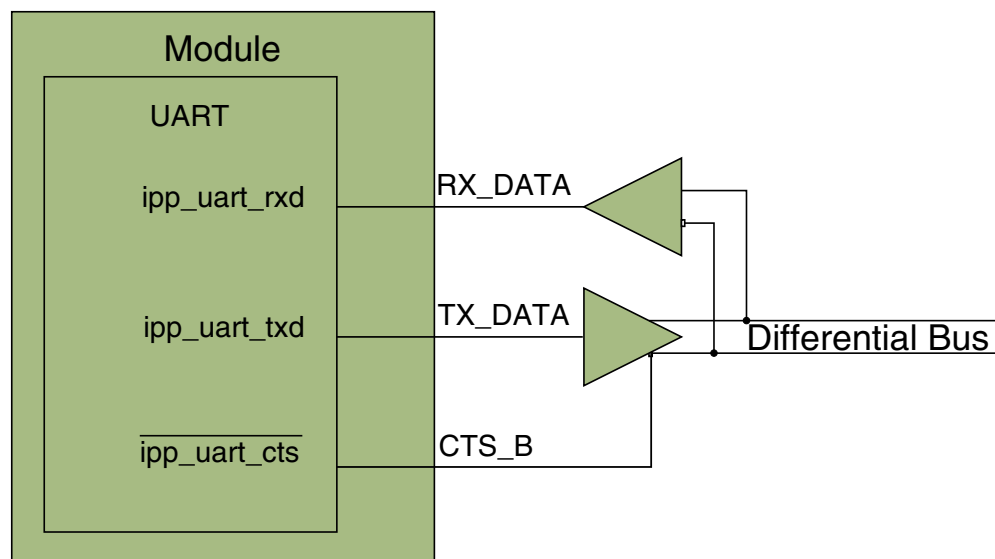


Figure 51-11. RS-485 driver connection (UART in DCE mode)

### 51.7.2 Transmit 9-bit RS-485 frames

To transmit 9-bit RS-485 frames, user need to enable parity (PREN=1) to enable trasmitting the ninth data bit, set 8-bit data word size (WS=1), and write TXB8 (UMCR[2]) as the 9<sup>th</sup> bit (bit [8]) to be transmitted (write '0' to TXB8 to transmit a data frame, write '1' to transmit a address frame). The other data bit [7:0] is written to TxFIFO by writing to the UTXD same as normal RS-232 operation.

### 51.7.3 Receive 9-bit RS-485 frames

To receive 9-bit RS-485 frames, user need to enable parity (PREN=1) to enable receiving the ninth data bit, set 8-bit data word size (WS=1). The receiver will save the 9-bit data to RxFIFO, and user should read the 9<sup>th</sup> databit (bit [8]) by reading the PRERR (URXD[10]) bit, and read data bit [7:0] by reading the RX\_DATA (URXD[7:0]).

There are two slave address detect modes, normal detect mode and automatic detect mode, and can be selected by SLAM (UMCR[1]).

### 51.7.3.1 RS-485 Slave Address Normal Detect Mode

To enable Normal Detect mode, clear SLAM (UMCR[1] to 0). The receiver ignores all data frames (9<sup>th</sup> bit = 0) until an address frame is received (9<sup>th</sup> bit = 1). At that time, the slave address detected (SAD = USR1[3]) bit is asserted and the *interrupt\_uart* interrupt is generated (if SADEN = UMCR[3] = 1). The address byte and subsequent bytes are all put into RxFIFO along with their 9<sup>th</sup> bit. The UART will also generate DMA request *dma\_req\_rx* when the RxFIFO reaches the selected threshold (controlled by RXTL) if receive ready DMA (RXDMAEN = UCR1[8]) request is enabled.

User should read the 9<sup>th</sup> databit (bit [8]) by reading the PRERR (URXD[10]) bit, and read data bit [7:0] by reading the RX\_DATA (URXD[7:0]).

In this mode, once the UART has detected a 9<sup>th</sup> bit is equal to '1', it will always save the subsequent frames to RxFIFO. So the software must decide whether the address and data in RxFIFO are needed or not.

### 51.7.3.2 RS-485 Slave Address Automatic Detect Mode

To enable Automatic Detect Mode, set SLAM (UMCR[1]) to 1. The receiver tries to detect an address byte (frame 9<sup>th</sup> bit = 1) that matches the programmed SLADDR (UMCR[15:8]) character. If the received byte is a data or an address byte that does not match the programmed SLADDR character, the receiver will discard these data.

Once the UART receives a matching address byte, it will assert the slave address detected (SAD = USR1[3]) bit and the *interrupt\_uart* interrupt will be generated (if SADEN = UMCR[3] = 1). The address byte and subsequent bytes are all put into RxFIFO along with their 9<sup>th</sup> bit. If receive ready DMA (RXDMAEN = UCR1[8]) request is enabled, the UART will also generate DMA request *dma\_req\_rx* when the RxFIFO reaches the selected threshold (controlled by RXTL).

If another address byte is received and this address byte does not match SLADDR character, the receiver will discard the address byte and subsequent data byte. If the address byte again matches SLADDR character, the receiver will put this address byte and subsequent data byte in the RxFIFO along with their 9<sup>th</sup> bit.

User should read the 9<sup>th</sup> databit (bit [8]) by reading the PRERR (URXD[10]) bit, and read data bit [7:0] by reading the RX\_DATA (URXD[7:0]).

See [Initialization](#) for 9-bit RS-485 programming guide.

## 51.8 Low Power Modes

These modes are controlled by the signals *doze\_req* and *stop\_req*. The control/status/data registers won't change when getting in/out of low power modes.

**Table 51-14. UART Low Power State Operation**

	Normal State ( <i>doze_req</i> = 1'b0 & <i>stop_req</i> = 1'b0)	Doze State ( <i>doze_req</i> = 1'b1)		Stop State ( <i>stop_req</i> = 1'b1)
		DOZE bit = 0	DOZE bit = 1	
UART-Clock	ON	ON	ON	OFF
UART Serial / IrDA	ON	ON	OFF	OFF

### 51.8.1 UART Operation in System Doze Mode

While in Doze State (when *doze\_req* input pin is set to 1'b1), the UART behavior depends on the DOZE (UCR1[1]) control bit.

While the DOZE bit is negated, the UART serial interface is enabled. While the system is in the Doze State, and the DOZE bit is asserted, the UART is disabled. If the Doze State is entered with the DOZE bit asserted while the UART serial interface was receiving or transmitting data, it will complete the receive/transmit of the current character and signal to the far-end transmitter/receiver to stop sending/receiving.

### 51.8.2 UART Operation in System Stop Mode

The internal baud rate clocks of the transmitter and receiver are gated off if the *stop\_req* signal to UART is asserted. Even though the clocks at the input of the UART continue to run during system Stop mode, the UART will not do any transmission or reception.

The following UART interrupts wake the ARM platform processor from STOP mode:

- RTS (RTSD)
- IrDA Asynchronous WAKE (AIRINT)
- Asynchronous WAKE (AWAKE)
- RI (RIDELT in DTE mode only)
- DCD (DCDDELT in DTE mode only)
- DTR (DTRD in DCE mode only)
- DSR (DTRD in DTE mode only)



When an asynchronous WAKE (awake) interrupt exits the ARM platform from STOP mode, make sure that a dummy character is sent first because the first character may not be received correctly.

### 51.8.3 Power Saving Method in UART

The RXEN (UCR2[1]), TXEN (UCR2[2]) and UARTEN (UCR1[0]) bits are set by the user and provide software control of low-power modes.

Setting the UARTEN (UCR1[0]) bit to 0 shuts off the receiver and transmitter logic and the associated clocks.

If the UART is used only in transmit mode, UARTEN and TXEN must be set to 1. If the UART is used only in receive mode, UARTEN and RXEN must be set to 1. Setting TXEN or RXEN to 0 allows to save a lot of power.

## 51.9 UART Operation in System Debug State

The bit UTS [11] controls whether the UART will respond to the input signal *debug\_req*, or whether it will continue to run as normal.

If the UART is programmed to respond to *debug\_req*:

1. The UART will halt all operations upon detecting the *debug\_req* input.
2. A transfer in progress, either to/from a core (using the IP Bus interface) or to/from an external device, will be completed before halting. This means a single byte/word transfer, not an entire FIFO. Reception of any further data from an external device will be disabled.
3. Internal registers will continue to be writable and readable using the IP Bus interface. A read will leave the contents unaffected.
4. The RX FIFO is affected in debug mode in the following way:
  - All writes into the RX FIFO are prevented.
  - The bit RXDBG (UTS[9]) is used to select the readability of the RX FIFO during debug mode:

RXDBG = 0: hold the read pointer at the location it had upon entering debug mode, and URXD register returns only the data value at that location, no matter how many reads attempted.

RXDBG = 1, selectable at any time: Allow to read the characters received in Rx FIFO. It will not be possible to re-read previously read locations, nor will it be possible to readjust the read pointer to the value it had prior to entering debug mode.

## 51.10 Reset

This section describes how to reset the block and explains special requirements related to reset.

### 51.10.1 Hardware reset

All of registers, FIFOs, state machines and sequential elements can be reset to their initial values by hardware reset or power on reset.

### 51.10.2 Software reset

The status registers USR1 and USR2, BRM registers UBIR and UBMIR, Tx FIFO and Rx FIFO, and transmitter and receiver state machines can be reset by software reset. Internal logic will remain the software reset signal at active for about 4 *module\_clock* cycles.

Programmer can follow the following software reset sequence:

1. Clear the SRST\_B bit (UCR2[0])
2. Wait for software reset complete: poll SOFTRST bit (UTS[0]) until it is 0.
3. Re-program baud rate registers: Re-write UBIR and UBMIR.

## 51.11 Transfer Error

The UART can generate a transfer error on the peripheral bus in the following cases:

- Core is writing into a read-only register.
- Core is accessing (read or write) an unused location within the assigned address space reserved to UART.

- Core is writing into UTXD register with transmit interface disabled (TXEN=0 or UARTEN=0)
- Core is reading URXD register with receive interface disabled (RXEN=0 or UARTEN=0)

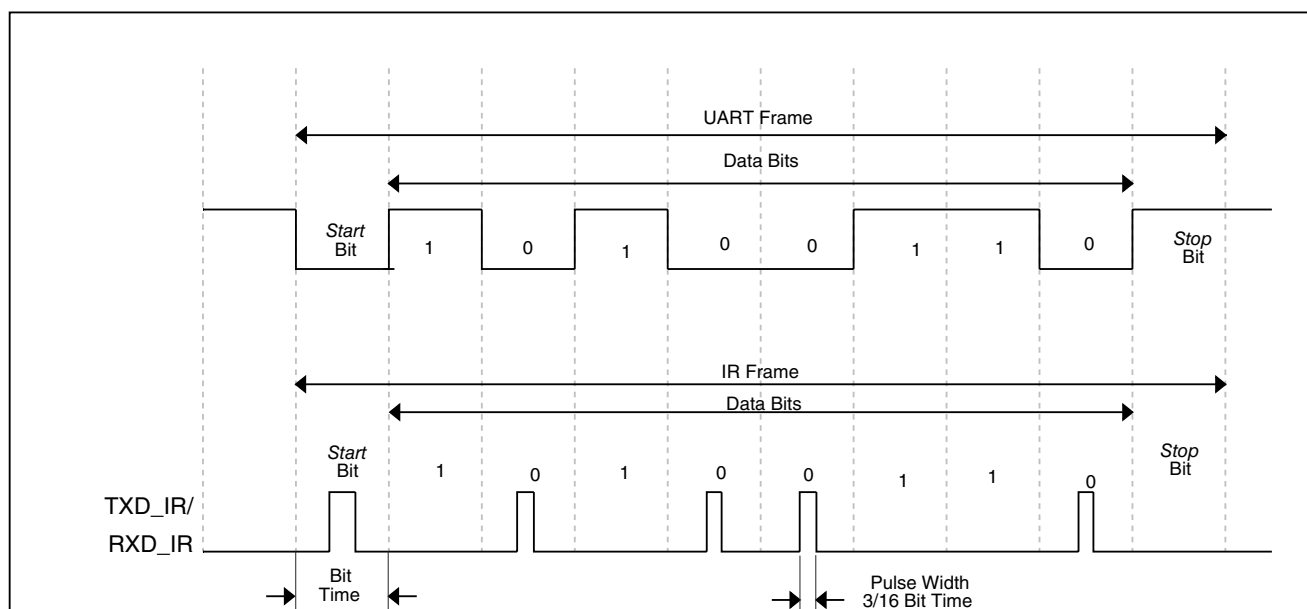
## 51.12 Functional Timing

This section includes timing diagrams for functional signaling.

### 51.12.1 IrDA Mode

According to IrDA specification, the low speed (115.2Kbit/s and below) IR frame format is compatible with UART frame.

In this figure, an example data 0x65 is used.



**Figure 51-12. Timing diagram of Low Speed IR (<=115.2 Kbit/s) Data Line**

## 51.13 Initialization

## 51.13.1 Programming the UART in RS-232 mode

As an example, the following sequence can be used to program the UART in order to send and receive characters in RS-232 mode.

Assumptions:

- Input uart clock = 100 MHz
- Baud rate = 921.6Kbps
- Data bits = 8 bits
- Parity = Even
- Stop bits = 1 bit
- Flow control = Hardware

Main program:

1. UCR1 = 0x0001

Enable the UART.

2. UCR2 = 0x2127

Set hardware flow control, data format and enable transmitter and receiver.

3. UCR3 = 0x0704

Set UCR3[RXDMUXSEL] = 1.

4. UCR4 = 0x7C00

Set CTS trigger level to 31,

5. UFCR = 0x089E

Set internal clock divider = 5 (divide input uart clock by 5). So the reference clock is  $100\text{MHz}/5 = 20\text{MHz}$ .

Set TXCTL = 2 and RXCTL = 30.

6. UBIR = 0x08FF

7. UBMCR = 0x0C34

In the above two steps, set baud rate to 921.6Kbps based on the 20MHz reference clock.

8. UCR1 = 0x2201

Enable the TRDY and RRDY interrupts.

9. UMCR = 0x0000

UMCR stay at default value 0x0000

Interrupt service routine for the transmitter:

- Write characters into UTXD

The TRDY interrupt will be automatically de-asserted when the data level of the TxFIFO exceeds the TXTL=2. Note: For the first time the interrupt may be de-asserted after 4 characters are written into the TxFIFO because of the shift register.

Interrupt service routine for the receiver:

- Read characters from URXD

The RRDY interrupt will be automatically de-asserted when the data level of the RxFIFO is below the RXTL=30.

### 51.13.2 Programming the UART in 9-bit RS-485 mode

As an example, the following sequence can be used to program the UART in order to send and receive frames in RS-485 mode.

Assumptions:

- Input uart clock = 100 MHz
- Baud rate = 5Mbps

Main program:

1. UCR1 = 0x0001

Enable the UART.

2. UCR2 = 0x4127

Set software flow control ( $\overline{\text{CTS}}$  pin is controlled by UCR2[12] ), enable parity(enable 9<sup>th</sup> bit rxd/txd), 8-bit word size , and enable transmitter and receiver.

3. UCR4 = 0x7C00

Set CTS trigger level to 31,

4. UFCR = 0x0A9E

Set RFDIV = 5 (divide input uart clock by 1), so the reference clock is 100MHz. Set UART in DCE mode (RS-485 driver connection outside the chip is the same as [Figure 51-11](#))

Set TXTL = 2 and RXTL = 30.

5. UBIR = 0x0003

6. UBMIR = 0x0004

In the above two steps, set baud rate to 5Mbps based on the 100MHz reference clock.

7. UCR1 = 0x2001 when UART as a master ,

or UCR1 = 0x0201 (or 0x0101) when UART as a slave.

Enable TRDY interrupt when UART as a master, enable RRDY interrupt or DMA request when UART as a slave.

8. UMCR = 0xA50B

Enable 9-bit RS-485 mode, enable SAD interrupt, set automatic slave address detect mode, set slave address is 0xA5.

Interrupt service routine for the transmitter:

- Transmit data: write its ninth bit (bit[8]) to UMCR[2], write its bit [7:0] into UTXD[7:0]

The TRDY interrupt will be automatically de-asserted when the data level of the TxFIFO exceeds the TXTL=2.

Note: For the first time the interrupt may be de-asserted after 4 characters are written into the TxFIFO because of the shift register.

Interrupt service routine for the receiver:

- Receive data: read its ninth bit (bit[8]) from URXD[10] , read its bit [7:0] from URXD[7:0].

Note: in RS-485 mode, URXD[10] bit is not the parity error, instead it holds the ninth bit (bit[8]) of the received data.

The SAD interrupt can not de-assert automatically, it needs MCU write 1 to USR1[3] to clear it . The RRDY interrupt or DMA request will be automatically de-asserted when the data level of the Rx FIFO is below the RXTL=30.

## 51.14 References

- EIA/TIA-232-F Interface Standard

<http://www.eia.org>, <http://www.tiaonline.org/standards>

- IrDA Standard

<http://www.irda.org>

## 51.15 UART Memory Map/Register Definition

UART supports 8-bit, 16-bit and 32-bit accesses to 32-bit memory-mapped addresses. Any access to unmapped memory location will yield a transfer error.

All registers except the ONEMS described in this section are 16-bit registers. The ONEMS register is a 24-bit register.

- For 32-bit write accesses, the upper two bytes will not be taken into account.
- For 32-bit read accesses the upper two bytes will return 0.

The ONEMS register is expanded from 16 bits to 24 bits in order to support the high frequency of the BRM internal clock *ref\_clk* (*module\_clock* after divider). The ONEMS register can be accessed as 8 bits, 16 bits or 32 bits.

- For 32-bit write accesses, the most significant byte of the ONEMS will be discarded.
- For 32-bit read accesses, the most significant byte of the ONEMS will be read as 0.

**UART memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
201_8000	UART Receiver Register (UART5_URXD)	32	R	0000_0000h	<a href="#">51.15.1/2976</a>
201_8040	UART Transmitter Register (UART5_UTXD)	32	W	0000_0000h	<a href="#">51.15.2/2978</a>
201_8080	UART Control Register 1 (UART5_UCR1)	32	R/W	0000_0000h	<a href="#">51.15.3/2979</a>
201_8084	UART Control Register 2 (UART5_UCR2)	32	R/W	0000_0001h	<a href="#">51.15.4/2981</a>
201_8088	UART Control Register 3 (UART5_UCR3)	32	R/W	0000_0700h	<a href="#">51.15.5/2984</a>
201_808C	UART Control Register 4 (UART5_UCR4)	32	R/W	0000_8000h	<a href="#">51.15.6/2986</a>
201_8090	UART FIFO Control Register (UART5_UFCR)	32	R/W	0000_0801h	<a href="#">51.15.7/2988</a>

*Table continues on the next page...*

## UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
201_8094	UART Status Register 1 (UART5_USR1)	32	R/W	0000_2040h	<a href="#">51.15.8/2990</a>
201_8098	UART Status Register 2 (UART5_USR2)	32	R/W	0000_4028h	<a href="#">51.15.9/2993</a>
201_809C	UART Escape Character Register (UART5_UESC)	32	R/W	0000_002Bh	<a href="#">51.15.10/2995</a>
201_80A0	UART Escape Timer Register (UART5_UTIM)	32	R/W	0000_0000h	<a href="#">51.15.11/2996</a>
201_80A4	UART BRM Incremental Register (UART5_UBIR)	32	R/W	0000_0000h	<a href="#">51.15.12/2996</a>
201_80A8	UART BRM Modulator Register (UART5_UBMR)	32	R/W	0000_0000h	<a href="#">51.15.13/2997</a>
201_80AC	UART Baud Rate Count Register (UART5_UBRC)	32	R	0000_0004h	<a href="#">51.15.14/2997</a>
201_80B0	UART One Millisecond Register (UART5_ONEMS)	32	R/W	0000_0000h	<a href="#">51.15.15/2998</a>
201_80B4	UART Test Register (UART5_UTS)	32	R/W	0000_0060h	<a href="#">51.15.16/2999</a>
201_80B8	UART RS-485 Mode Control Register (UART5_UMCR)	32	R/W	0000_0000h	<a href="#">51.15.17/3000</a>
202_0000	UART Receiver Register (UART1_URXD)	32	R	0000_0000h	<a href="#">51.15.1/2976</a>
202_0040	UART Transmitter Register (UART1_UTXD)	32	W	0000_0000h	<a href="#">51.15.2/2978</a>
202_0080	UART Control Register 1 (UART1_UCR1)	32	R/W	0000_0000h	<a href="#">51.15.3/2979</a>
202_0084	UART Control Register 2 (UART1_UCR2)	32	R/W	0000_0001h	<a href="#">51.15.4/2981</a>
202_0088	UART Control Register 3 (UART1_UCR3)	32	R/W	0000_0700h	<a href="#">51.15.5/2984</a>
202_008C	UART Control Register 4 (UART1_UCR4)	32	R/W	0000_8000h	<a href="#">51.15.6/2986</a>
202_0090	UART FIFO Control Register (UART1_UFCR)	32	R/W	0000_0801h	<a href="#">51.15.7/2988</a>
202_0094	UART Status Register 1 (UART1_USR1)	32	R/W	0000_2040h	<a href="#">51.15.8/2990</a>
202_0098	UART Status Register 2 (UART1_USR2)	32	R/W	0000_4028h	<a href="#">51.15.9/2993</a>
202_009C	UART Escape Character Register (UART1_UESC)	32	R/W	0000_002Bh	<a href="#">51.15.10/2995</a>
202_00A0	UART Escape Timer Register (UART1_UTIM)	32	R/W	0000_0000h	<a href="#">51.15.11/2996</a>
202_00A4	UART BRM Incremental Register (UART1_UBIR)	32	R/W	0000_0000h	<a href="#">51.15.12/2996</a>

Table continues on the next page...



**UART memory map (continued)**

<b>Absolute address (hex)</b>	<b>Register name</b>	<b>Width (in bits)</b>	<b>Access</b>	<b>Reset value</b>	<b>Section/page</b>
202_00A8	UART BRM Modulator Register (UART1_UBMR)	32	R/W	0000_0000h	<a href="#">51.15.13/2997</a>
202_00AC	UART Baud Rate Count Register (UART1_UBRC)	32	R	0000_0004h	<a href="#">51.15.14/2997</a>
202_00B0	UART One Millisecond Register (UART1_ONEMS)	32	R/W	0000_0000h	<a href="#">51.15.15/2998</a>
202_00B4	UART Test Register (UART1_UTS)	32	R/W	0000_0060h	<a href="#">51.15.16/2999</a>
202_00B8	UART RS-485 Mode Control Register (UART1_UMCR)	32	R/W	0000_0000h	<a href="#">51.15.17/3000</a>
202_4000	UART Receiver Register (UART2_URXD)	32	R	0000_0000h	<a href="#">51.15.1/2976</a>
202_4040	UART Transmitter Register (UART2_UTXD)	32	W	0000_0000h	<a href="#">51.15.2/2978</a>
202_4080	UART Control Register 1 (UART2_UCR1)	32	R/W	0000_0000h	<a href="#">51.15.3/2979</a>
202_4084	UART Control Register 2 (UART2_UCR2)	32	R/W	0000_0001h	<a href="#">51.15.4/2981</a>
202_4088	UART Control Register 3 (UART2_UCR3)	32	R/W	0000_0700h	<a href="#">51.15.5/2984</a>
202_408C	UART Control Register 4 (UART2_UCR4)	32	R/W	0000_8000h	<a href="#">51.15.6/2986</a>
202_4090	UART FIFO Control Register (UART2_UFCR)	32	R/W	0000_0801h	<a href="#">51.15.7/2988</a>
202_4094	UART Status Register 1 (UART2_USR1)	32	R/W	0000_2040h	<a href="#">51.15.8/2990</a>
202_4098	UART Status Register 2 (UART2_USR2)	32	R/W	0000_4028h	<a href="#">51.15.9/2993</a>
202_409C	UART Escape Character Register (UART2_UESC)	32	R/W	0000_002Bh	<a href="#">51.15.10/2995</a>
202_40A0	UART Escape Timer Register (UART2_UTIM)	32	R/W	0000_0000h	<a href="#">51.15.11/2996</a>
202_40A4	UART BRM Incremental Register (UART2_UBIR)	32	R/W	0000_0000h	<a href="#">51.15.12/2996</a>
202_40A8	UART BRM Modulator Register (UART2_UBMR)	32	R/W	0000_0000h	<a href="#">51.15.13/2997</a>
202_40AC	UART Baud Rate Count Register (UART2_UBRC)	32	R	0000_0004h	<a href="#">51.15.14/2997</a>
202_40B0	UART One Millisecond Register (UART2_ONEMS)	32	R/W	0000_0000h	<a href="#">51.15.15/2998</a>
202_40B4	UART Test Register (UART2_UTS)	32	R/W	0000_0060h	<a href="#">51.15.16/2999</a>
202_40B8	UART RS-485 Mode Control Register (UART2_UMCR)	32	R/W	0000_0000h	<a href="#">51.15.17/3000</a>

## UART memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
203_4000	UART Receiver Register (UART3_URXD)	32	R	0000_0000h	<a href="#">51.15.1/2976</a>
203_4040	UART Transmitter Register (UART3_UTXD)	32	W	0000_0000h	<a href="#">51.15.2/2978</a>
203_4080	UART Control Register 1 (UART3_UCR1)	32	R/W	0000_0000h	<a href="#">51.15.3/2979</a>
203_4084	UART Control Register 2 (UART3_UCR2)	32	R/W	0000_0001h	<a href="#">51.15.4/2981</a>
203_4088	UART Control Register 3 (UART3_UCR3)	32	R/W	0000_0700h	<a href="#">51.15.5/2984</a>
203_408C	UART Control Register 4 (UART3_UCR4)	32	R/W	0000_8000h	<a href="#">51.15.6/2986</a>
203_4090	UART FIFO Control Register (UART3_UFCR)	32	R/W	0000_0801h	<a href="#">51.15.7/2988</a>
203_4094	UART Status Register 1 (UART3_USR1)	32	R/W	0000_2040h	<a href="#">51.15.8/2990</a>
203_4098	UART Status Register 2 (UART3_USR2)	32	R/W	0000_4028h	<a href="#">51.15.9/2993</a>
203_409C	UART Escape Character Register (UART3_UESC)	32	R/W	0000_002Bh	<a href="#">51.15.10/2995</a>
203_40A0	UART Escape Timer Register (UART3_UTIM)	32	R/W	0000_0000h	<a href="#">51.15.11/2996</a>
203_40A4	UART BRM Incremental Register (UART3_UBIR)	32	R/W	0000_0000h	<a href="#">51.15.12/2996</a>
203_40A8	UART BRM Modulator Register (UART3_UBMR)	32	R/W	0000_0000h	<a href="#">51.15.13/2997</a>
203_40AC	UART Baud Rate Count Register (UART3_UBRC)	32	R	0000_0004h	<a href="#">51.15.14/2997</a>
203_40B0	UART One Millisecond Register (UART3_ONEMS)	32	R/W	0000_0000h	<a href="#">51.15.15/2998</a>
203_40B4	UART Test Register (UART3_UTS)	32	R/W	0000_0060h	<a href="#">51.15.16/2999</a>
203_40B8	UART RS-485 Mode Control Register (UART3_UMCR)	32	R/W	0000_0000h	<a href="#">51.15.17/3000</a>
203_8000	UART Receiver Register (UART4_URXD)	32	R	0000_0000h	<a href="#">51.15.1/2976</a>
203_8040	UART Transmitter Register (UART4_UTXD)	32	W	0000_0000h	<a href="#">51.15.2/2978</a>
203_8080	UART Control Register 1 (UART4_UCR1)	32	R/W	0000_0000h	<a href="#">51.15.3/2979</a>
203_8084	UART Control Register 2 (UART4_UCR2)	32	R/W	0000_0001h	<a href="#">51.15.4/2981</a>
203_8088	UART Control Register 3 (UART4_UCR3)	32	R/W	0000_0700h	<a href="#">51.15.5/2984</a>

Table continues on the next page...

## UART memory map (continued)

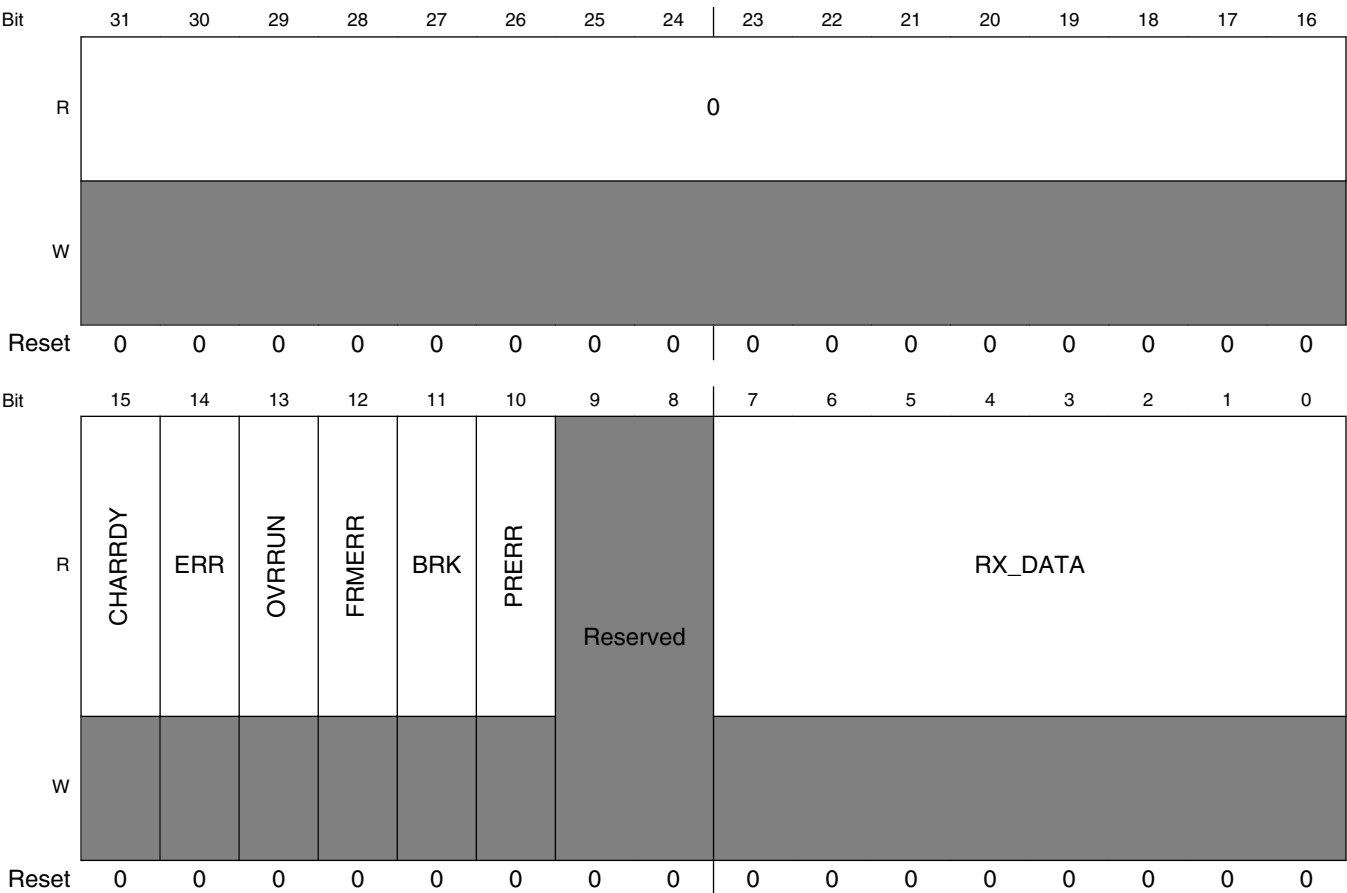
Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
203_808C	UART Control Register 4 (UART4_UCR4)	32	R/W	0000_8000h	<a href="#">51.15.6/2986</a>
203_8090	UART FIFO Control Register (UART4_UFCR)	32	R/W	0000_0801h	<a href="#">51.15.7/2988</a>
203_8094	UART Status Register 1 (UART4_USR1)	32	R/W	0000_2040h	<a href="#">51.15.8/2990</a>
203_8098	UART Status Register 2 (UART4_USR2)	32	R/W	0000_4028h	<a href="#">51.15.9/2993</a>
203_809C	UART Escape Character Register (UART4_UESC)	32	R/W	0000_002Bh	<a href="#">51.15.10/2995</a>
203_80A0	UART Escape Timer Register (UART4_UTIM)	32	R/W	0000_0000h	<a href="#">51.15.11/2996</a>
203_80A4	UART BRM Incremental Register (UART4_UBIR)	32	R/W	0000_0000h	<a href="#">51.15.12/2996</a>
203_80A8	UART BRM Modulator Register (UART4_UBMR)	32	R/W	0000_0000h	<a href="#">51.15.13/2997</a>
203_80AC	UART Baud Rate Count Register (UART4_UBRC)	32	R	0000_0004h	<a href="#">51.15.14/2997</a>
203_80B0	UART One Millisecond Register (UART4_ONEMS)	32	R/W	0000_0000h	<a href="#">51.15.15/2998</a>
203_80B4	UART Test Register (UART4_UTS)	32	R/W	0000_0060h	<a href="#">51.15.16/2999</a>
203_80B8	UART RS-485 Mode Control Register (UART4_UMCR)	32	R/W	0000_0000h	<a href="#">51.15.17/3000</a>

51.15.2 UART Receiver Register (UARTx\_URXD)

NOTE

The UART will yield a transfer error on the peripheral bus when core is reading URXD register with receive interface disabled (RXEN=0 or UARTEN=0).

Address: Base address + 0h offset



UARTx\_URXD field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15 CHARRDY	Character Ready. This read-only bit indicates an invalid read when the FIFO becomes empty and software tries to read the same old data. This bit should not be used for polling for data written to the RX FIFO.  0 Character in RX_DATA field and associated flags are invalid. 1 Character in RX_DATA field and associated flags valid and ready for reading.

Table continues on the next page...

## UARTx\_URXD field descriptions (continued)

Field	Description
14 ERR	<p><b>Error Detect.</b> Indicates whether the character present in the RX_DATA field has an error (OVRUN, FRMERR, BRK or PRERR) status. The ERR bit is updated and valid for each received character.</p> <p>0 No error status was detected 1 An error status was detected</p>
13 OVRUN	<p><b>Receiver Overrun.</b> This read-only bit, when HIGH, indicates that the corresponding character was stored in the last position (32nd) of the Rx FIFO. Even if a 33rd character has not been detected, this bit will be set to '1' for the 32nd character.</p> <p>0 No RxFIFO overrun was detected 1 A RxFIFO overrun was detected</p>
12 FRMERR	<p><b>Frame Error.</b> Indicates whether the current character had a framing error (a missing stop bit) and is possibly corrupted. FRMERR is updated for each character read from the RxFIFO.</p> <p>0 The current character has no framing error 1 The current character has a framing error</p>
11 BRK	<p><b>BREAK Detect.</b> Indicates whether the current character was detected as a BREAK character. The data bits and the stop bit are all 0. The FRMERR bit is set when BRK is set. When odd parity is selected, PRERR is also set when BRK is set. BRK is valid for each character read from the RxFIFO.</p> <p>0 The current character is not a BREAK character 1 The current character is a BREAK character</p>
10 PRERR	<p><b>In RS-485 mode, it holds the ninth data bit (bit [8]) of received 9-bit RS-485 data</b></p> <p><b>In RS232/IrDA mode, it is the Parity Error flag.</b> Indicates whether the current character was detected with a parity error and is possibly corrupted. PRERR is updated for each character read from the RxFIFO. When parity is disabled, PRERR always reads as 0.</p> <p>0 = No parity error was detected for data in the RX_DATA field 1 = A parity error was detected for data in the RX_DATA field</p>
9–8 -	<p>This field is reserved.</p> <p><b>Reserved</b></p>
7–0 RX_DATA	<p><b>Received Data.</b> Holds the received character. In 7-bit mode, the most significant bit (MSB) is forced to 0. In 8-bit mode, all bits are active.</p>

### 51.15.3 UART Transmitter Register (UARTx\_UTXD)

**NOTE**

The UART will yield a transfer error on the peripheral bus when core is writing into UART\_URXD register with transmit interface disabled (TXEN=0 or UARTEN=0).

Memory space between UART\_URXD and UART\_UTXD registers is reserved. Any read or write access to this space will be considered as an invalid access and yield a transfer error.

Address: Base address + 40h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																0															
W																									TX_DATA							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**UARTx\_UTXD field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–8 Reserved	This read-only field is reserved and always has the value 0.
7–0 TX_DATA	<b>Transmit Data.</b> Holds the parallel transmit data inputs. In 7-bit mode, D7 is ignored. In 8-bit mode, all bits are used. Data is transmitted least significant bit (LSB) first. A new character is transmitted when the TX_DATA field is written. The TX_DATA field must be written only when the TRDY bit is high to ensure that corrupted data is not sent.

## 51.15.4 UART Control Register 1 (UARTx\_UCR1)

Address: Base address + 80h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADEN	ADBR	TRDYEN	IDEN	ICD	RRDYEN	RXDMAEN	IREN	TXEMPTYEN	RTSDEN	SNDBRK	TXDMAEN	ATDMAEN	DOZE	UARTEN	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### UARTx\_UCR1 field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15 ADEN	<b>Automatic Baud Rate Detection Interrupt Enable.</b> Enables/Disables the automatic baud rate detect complete (ADET) bit to generate an interrupt ( <i>interrupt_uart</i> = 0).  0 Disable the automatic baud rate detection interrupt 1 Enable the automatic baud rate detection interrupt
14 ADBR	<b>Automatic Detection of Baud Rate.</b> Enables/Disables automatic baud rate detection. When the ADBR bit is set and the ADET bit is cleared, the receiver detects the incoming baud rate automatically. The ADET flag is set when the receiver verifies that the incoming baud rate is detected properly by detecting an ASCII character "A" or "a" (0x41 or 0x61).  0 Disable automatic detection of baud rate 1 Enable automatic detection of baud rate
13 TRDYEN	<b>Transmitter Ready Interrupt Enable.</b> Enables/Disables the transmitter Ready Interrupt (TRDY) when the transmitter has one or more slots available in the TxFIFO. The fill level in the TxFIFO at which an interrupt is generated is controlled by TxTL bits. When TRDYEN is negated, the transmitter ready interrupt is disabled.  <b>NOTE:</b> An interrupt will be issued as long as TRDYEN and TRDY are high even if the transmitter is not enabled. In general, user should enable the transmitter before enabling the TRDY interrupt.  0 Disable the transmitter ready interrupt 1 Enable the transmitter ready interrupt
12 IDEN	<b>Idle Condition Detected Interrupt Enable.</b> Enables/Disables the IDLE bit to generate an interrupt ( <i>interrupt_uart</i> = 0).  0 Disable the IDLE interrupt 1 Enable the IDLE interrupt

Table continues on the next page...

## UARTx\_UCR1 field descriptions (continued)

Field	Description
11–10 ICD	<b>Idle Condition Detect.</b> Controls the number of frames RXD is allowed to be idle before an idle condition is reported.  00 Idle for more than 4 frames 01 Idle for more than 8 frames 10 Idle for more than 16 frames 11 Idle for more than 32 frames
9 RRDYEN	<b>Receiver Ready Interrupt Enable.</b> Enables/Disables the RRDY interrupt when the RxFIFO contains data. The fill level in the RxFIFO at which an interrupt is generated is controlled by the RXTL bits. When RRDYEN is negated, the receiver ready interrupt is disabled.  0 Disables the RRDY interrupt 1 Enables the RRDY interrupt
8 RXDMAEN	<b>Receive Ready DMA Enable.</b> Enables/Disables the receive DMA request <i>dma_req_rx</i> when the receiver has data in the RxFIFO. The fill level in the RxFIFO at which a DMA request is generated is controlled by the RXTL bits. When negated, the receive DMA request is disabled.  0 Disable DMA request 1 Enable DMA request
7 IREN	<b>Infrared Interface Enable.</b> Enables/Disables the IR interface. See the IR interface description in <a href="#">Infrared Interface</a> , for more information.  Note: MDEN(UMCR[0]) must be cleared to 0 when using IrDA interface. See <a href="#">Table 51-1</a>  0 Disable the IR interface 1 Enable the IR interface
6 TXMPTYEN	<b>Transmitter Empty Interrupt Enable.</b> Enables/Disables the transmitter FIFO empty (TXFE) interrupt. <i>interrupt_uart</i> . When negated, the TXFE interrupt is disabled.  <b>NOTE:</b> An interrupt will be issued as long as TXMPTYEN and TXFE are high even if the transmitter is not enabled. In general, user should enable the transmitter before enabling the TXFE interrupt.  0 Disable the transmitter FIFO empty interrupt 1 Enable the transmitter FIFO empty interrupt
5 RTSDEN	<b>RTS Delta Interrupt Enable.</b> Enables/Disables the RTSD interrupt. The current status of the RTS_B pin is read in the RTSS bit.  0 Disable RTSD interrupt 1 Enable RTSD interrupt
4 SNDBRK	<b>Send BREAK.</b> Forces the transmitter to send a BREAK character. The transmitter finishes sending the character in progress (if any) and sends BREAK characters until SNDBRK is reset. Because the transmitter samples SNDBRK after every bit is transmitted, it is important that SNDBRK is asserted high for a sufficient period of time to generate a valid BREAK. After the BREAK transmission completes, the UART transmits 2 mark bits. The user can continue to fill the TxFIFO and any characters remaining are transmitted when the BREAK is terminated.  0 Do not send a BREAK character 1 Send a BREAK character (continuous 0s)
3 TXDMAEN	<b>Transmitter Ready DMA Enable.</b> Enables/Disables the transmit DMA request <i>dma_req_tx</i> when the transmitter has one or more slots available in the TxFIFO. The fill level in the TxFIFO that generates the <i>dma_req_tx</i> is controlled by the TXTL bits.

Table continues on the next page...



## UARTx\_UCR1 field descriptions (continued)

Field	Description
	<p><b>NOTE:</b> A DMA request will be issued as long as TXDMAEN and TRDY are high even if the transmitter is not enabled. In general, user should enable the transmitter before enabling the transmit DMA request.</p> <p>0 Disable transmit DMA request 1 Enable transmit DMA request</p>
2 ATDMAEN	<p><b>Aging DMA Timer Enable.</b> Enables/Disables the receive DMA request <i>dma_req_rx</i> for the aging timer interrupt (triggered with AGTIM flag in USR1[8]).</p> <p>0 Disable AGTIM DMA request 1 Enable AGTIM DMA request</p>
1 DOZE	<p><b>DOZE.</b> Determines the UART enable condition in the DOZE state. When <i>doze_req</i> input pin is at '1', (the ARM Platform executes a doze instruction and the system is placed in the Doze State), the DOZE bit affects operation of the UART. While in the Doze State, if this bit is asserted, the UART is disabled. See the description in <a href="#">Low Power Modes</a>.</p> <p>0 The UART is enabled when in DOZE state 1 The UART is disabled when in DOZE state</p>
0 UARTEN	<p><b>UART Enable.</b> Enables/Disables the UART. If UARTEN is negated in the middle of a transmission, the transmitter stops and pulls the TXD line to a logic 1. UARTEN must be set to 1 before any access to UTXD and URXD registers, otherwise a transfer error is returned.</p> <p>This bit can be set to 1 along with other bits in this register. There is no restriction to the sequence of programing this bit and other control registers.</p> <p>0 Disable the UART 1 Enable the UART</p>

## 51.15.5 UART Control Register 2 (UARTx\_UCR2)

Address: Base address + 84h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ESCI	IRTS	CTSC	CTS	ESSEN	RTEC	PREN	PRO E	STPB	WS	RTSEN	ATEN	TXEN	RXEN	SRST	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

## UARTx\_UCR2 field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15 ESCI	<b>Escape Sequence Interrupt Enable.</b> Enables/Disables the ESCF bit to generate an interrupt.  0 Disable the escape sequence interrupt 1 Enable the escape sequence interrupt
14 IRTS	<b>Ignore RTS Pin.</b> Forces the RTS input signal presented to the transmitter to always be asserted (set to low), effectively ignoring the external pin. When in this mode, the RTS pin serves as a general purpose input.  0 Transmit only when the RTS pin is asserted 1 Ignore the RTS pin
13 CTSC	<b>CTS Pin Control.</b> Controls the operation of the CTS_B output pin. When CTSC is asserted, the CTS_B output pin is controlled by the receiver. When the RxFIFO is filled to the level of the programmed trigger level and the start bit of the overflowing character (TRIGGER LEVEL + 1) is validated, the CTS_B output pin is negated to indicate to the far-end transmitter to stop transmitting. When the trigger level is programmed for less than 32, the receiver continues to receive data until the RxFIFO is full. When the CTSC bit is negated, the CTS_B output pin is controlled by the CTS bit. On reset, because CTSC is cleared to 0, the CTS_B pin is controlled by the CTS bit, which again is cleared to 0 on reset. This means that on reset the CTS_B signal is negated.  0 The CTS_B pin is controlled by the CTS bit 1 The CTS_B pin is controlled by the receiver
12 CTS	<b>Clear to Send.</b> Controls the CTS_B pin when the CTSC bit is negated. CTS has no function when CTSC is asserted.  0 The CTS_B pin is high (inactive) 1 The CTS_B pin is low (active)
11 ESCEN	<b>Escape Enable.</b> Enables/Disables the escape sequence detection logic.  0 Disable escape sequence detection 1 Enable escape sequence detection
10–9 RTEC	<b>Request to Send Edge Control.</b> Selects the edge that triggers the RTS interrupt. This has no effect on the RTS delta interrupt. RTEC has an effect only when RTSEN = 1 (see <a href="#">Table 51-8</a> ).  00 Trigger interrupt on a rising edge 01 Trigger interrupt on a falling edge 1X Trigger interrupt on any edge
8 PREN	<b>Parity Enable.</b> Enables/Disables the parity generator in the transmitter and parity checker in the receiver. When PREN is asserted, the parity generator and checker are enabled, and disabled when PREN is negated.  0 Disable parity generator and checker 1 Enable parity generator and checker
7 PROE	<b>Parity Odd/Even.</b> Controls the sense of the parity generator and checker. When PROE is high, odd parity is generated and expected. When PROE is low, even parity is generated and expected. PROE has no function if PREN is low.  0 Even parity 1 Odd parity

Table continues on the next page...

## UARTx\_UCR2 field descriptions (continued)

Field	Description
6 STPB	<p><b>Stop.</b> Controls the number of stop bits after a character. When STPB is low, 1 stop bit is sent. When STPB is high, 2 stop bits are sent. STPB also affects the receiver.</p> <p>0 The transmitter sends 1 stop bit. The receiver expects 1 or more stop bits.</p> <p>1 The transmitter sends 2 stop bits. The receiver expects 2 or more stop bits.</p>
5 WS	<p><b>Word Size.</b> Controls the character length. When WS is high, the transmitter and receiver are in 8-bit mode. When WS is low, they are in 7-bit mode. The transmitter ignores bit 7 and the receiver sets bit 7 to 0. WS can be changed in-between transmission (reception) of characters, however not when a transmission (reception) is in progress, in which case the length of the current character being transmitted (received) is unpredictable.</p> <p>0 7-bit transmit and receive character length (not including START, STOP or PARITY bits)</p> <p>1 8-bit transmit and receive character length (not including START, STOP or PARITY bits)</p>
4 RTSEN	<p><b>Request to Send Interrupt Enable.</b> Controls the RTS edge sensitive interrupt. When RTSEN is asserted and the programmed edge is detected on the RTS_B pin (the RTSF bit is asserted), an interrupt will be generated on the <i>interrupt_uart</i> pin. (See <a href="#">Table 51-8.</a>)</p> <p>0 Disable request to send interrupt</p> <p>1 Enable request to send interrupt</p>
3 ATEN	<p><b>Aging Timer Enable.</b> This bit is used to enable the aging timer interrupt (triggered with AGTIM)</p> <p>0 AGTIM interrupt disabled</p> <p>1 AGTIM interrupt enabled</p>
2 TXEN	<p><b>Transmitter Enable.</b> Enables/Disables the transmitter. When TXEN is negated the transmitter is disabled and idle. When the UARTEN and TXEN bits are set the transmitter is enabled. If TXEN is negated in the middle of a transmission, the UART disables the transmitter immediately, and starts marking 1s. The transmitter FIFO cannot be written when this bit is cleared.</p> <p>0 Disable the transmitter</p> <p>1 Enable the transmitter</p>
1 RXEN	<p><b>Receiver Enable.</b> Enables/Disables the receiver. When the receiver is enabled, if the RXD input is already low, the receiver does not recognize BREAK characters, because it requires a valid 1-to-0 transition before it can accept any character.</p> <p>0 Disable the receiver</p> <p>1 Enable the receiver</p>
0 SRST	<p><b>Software Reset.</b> Once the software writes 0 to SRST_B, the software reset remains active for 4 <i>module_clock</i> cycles before the hardware deasserts SRST_B. The software can only write 0 to SRST_B. Writing 1 to SRST_B is ignored.</p> <p>0 Reset the transmit and receive state machines, all FIFOs and register USR1, USR2, UBIR, UBMR, UBRC, URXD, UTXD and UTS[6-3].</p> <p>1 No reset</p>

## 51.15.6 UART Control Register 3 (UARTx\_UCR3)

Address: Base address + 88h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DPEC		DTREN	PARERREN	FRAERREN	DSR	DCD	RI	ADNIMP	RXDSEN	AIRINTEN	AWAKEN	DTRDEN	RXDMUXSEL	INVT	ACIEN
W																
Reset	0	0	0	0	0	1	1	1	0	0	0	0	0	0	0	0

### UARTx\_UCR3 field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–14 DPEC	<b>DTR/DSR Interrupt Edge Control.</b> These bits control the edge of DTR_B (DCE) or DSR_B (DTE) on which an interrupt will be generated. An interrupt will only be generated if the DTREN bit is set.  00 interrupt generated on rising edge 01 interrupt generated on falling edge 1X interrupt generated on either edge
13 DTREN	<b>Data Terminal Ready Interrupt Enable.</b> When this bit is set, it will enable the status bit DTRF (USR2 [13]) (DTR/DSR edge sensitive interrupt) to cause an interrupt.  0 Data Terminal Ready Interrupt Disabled 1 Data Terminal Ready Interrupt Enabled
12 PARERREN	<b>Parity Error Interrupt Enable. Enables/Disables the interrupt. When asserted, PARERREN causes the PARITYERR bit to generate an interrupt.</b>  0 Disable the parity error interrupt 1 Enable the parity error interrupt
11 FRAERREN	<b>Frame Error Interrupt Enable. Enables/Disables the interrupt. When asserted, FRAERREN causes the FRAMERR bit to generate an interrupt.</b>  0 Disable the frame error interrupt 1 Enable the frame error interrupt
10 DSR	<b>Data Set Ready.</b> This bit is used by software to control the DSR/DTR output pin for the modem interface. In DCE mode it applies to DSR_B and in DTE mode it applies to DTR_B.  0 DSR/ DTR pin is logic zero 1 DSR/ DTR pin is logic one

Table continues on the next page...

## UARTx\_UCR3 field descriptions (continued)

Field	Description
9 DCD	<p><b>Data Carrier Detect.</b> In DCE mode this bit is used by software to control the DCD_B output pin for the modem interface. In DTE mode, when this bit is set, it will enable the status bit DCDELT (USR2 (6)) to cause an interrupt.</p> <p>0 DCD_B pin is logic zero (DCE mode)  1 DCD_B pin is logic one (DCE mode)  0 DCDELT interrupt disabled (DTE mode)  1 DCDELT interrupt enabled (DTE mode)</p>
8 RI	<p><b>Ring Indicator.</b> In DCE mode this bit is used by software to control the RI_B output pin for the modem interface. In DTE mode, when this bit is set, it will enable the status bit RIDELT (USR2 (10)) to cause an interrupt.</p> <p>0 RI_B pin is logic zero (DCE mode)  1 RI_B pin is logic one (DCE mode)  0 RIDELT interrupt disabled (DTE mode)  1 RIDELT interrupt enabled (DTE mode)</p>
7 ADNIMP	<p><b>Autobaud Detection Not Improved- Disables new features of autobaud detection (See <a href="#">Baud Rate Automatic Detection Protocol</a>, for more details).</b></p> <p>0 Autobaud detection new features selected  1 Keep old autobaud detection mechanism</p>
6 RXDSEN	<p>Receive Status Interrupt Enable. Controls the receive status interrupt (<i>interrupt_uart</i>). When this bit is enabled and RXDS status bit is set, the interrupt <i>interrupt_uart</i> will be generated.</p> <p>0 Disable the RXDS interrupt  1 Enable the RXDS interrupt</p>
5 AIRINTEN	<p>Asynchronous IR WAKE Interrupt Enable. Controls the asynchronous IR WAKE interrupt. An interrupt is generated when AIRINTEN is asserted and a pulse is detected on the RXD pin.</p> <p>0 Disable the AIRINT interrupt  1 Enable the AIRINT interrupt</p>
4 AWAKEN	<p>Asynchronous WAKE Interrupt Enable. Controls the asynchronous WAKE interrupt. An interrupt is generated when AWAKEN is asserted and a falling edge is detected on the RXD pin.</p> <p>0 Disable the AWAKE interrupt  1 Enable the AWAKE interrupt</p>
3 DTRDEN	<p><b>Data Terminal Ready Delta Enable.</b> Enables / Disables the asynchronous DTRD interrupt. When DTRDEN is asserted and an edge (rising or falling) is detected on DTR_B (in DCE mode) or on DSR_B (in DTE mode), then an interrupt is generated.</p> <p>0 Disable DTRD interrupt  1 Enable DTRD interrupt</p>
2 RXDMUXSEL	<p>RXD Muxed Input Selected. Selects proper input pins for serial and Infrared input signal.</p> <p><b>NOTE:</b> In this chip, UARTs are used in MUXED mode, so that this bit should always be set.</p>
1 INVT	<p>Invert TXD output in RS-232/RS-485 mode, set TXD active level in IrDA mode.</p> <p>In RS232/RS-485 mode(UMCR[0] = 1), if this bit is set to 1, the TXD output is inverted before transmitted.</p> <p>In <b>IrDA mode</b>, when INVT is cleared, the infrared logic block transmits a positive IR 3/16 pulse for all 0s and 0s are transmitted for 1s. When INVT is set (INVT = 1), the infrared logic block transmits an active low or negative infrared 3/16 pulse for all 0s and 1s are transmitted for 1s.</p>

Table continues on the next page...

## UARTx\_UCR3 field descriptions (continued)

Field	Description
	0 <b>TXD is not inverted</b> 1 <b>TXD is inverted</b> 0 <b>TXD</b> Active low transmission 1 <b>TXD</b> Active high transmission
0 ACIEN	<b>Autobaud Counter Interrupt Enable.</b> This bit is used to enable the autobaud counter stopped interrupt (triggered with ACST (USR2[11]). 0 ACST interrupt disabled 1 ACST interrupt enabled

## 51.15.7 UART Control Register 4 (UARTx\_UCR4)

Address: Base address + 8Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	0																
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	CTSTL							INVR	ENIRI	WKEN	IDDMAEN	IRSC	LPBYP	TCEN	BKEN	OREN	DREN
W																	
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

## UARTx\_UCR4 field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–10 CTSTL	<b>CTS Trigger Level.</b> Controls the threshold at which the CTS_B pin is deasserted by the RxFIFO. After the trigger level is reached and the CTS_B pin is deasserted, the RxFIFO continues to receive data until it is full. The CTSTL bits are encoded as shown in the Settings column. Settings 0 to 32 are in use. All other settings are Reserved. 000000 0 characters received 000001 1 characters in the RxFIFO ... — ... — 100000 32 characters in the RxFIFO (maximum)
9 INVR	<b>Invert RXD input in RS-232/RS-485 Mode, determine RXD input</b> logic level being sampled in <b>In IrDA mode</b> .

Table continues on the next page...

## UARTx\_UCR4 field descriptions (continued)

Field	Description
	<p><b>In RS232/RS-485 Mode(UMCR[0] = 1), if this bit is set to 1, the RXD input is inverted before sampled.</b></p> <p><b>In IrDA mode</b>,when cleared, the infrared logic block expects an active low or negative IR 3/16 pulse for 0s and 1s are expected for 1s. When INVR is set (INVR 1), the infrared logic block expects an active high or positive IR 3/16 pulse for 0s and 0s are expected for 1s.</p> <p>0 <b>RXD input is not inverted</b>  1 <b>RXD input is inverted</b>  0 <b>RXD</b> active low detection  1 <b>RXD</b> active high detection</p>
8 ENIRI	<p><b>Serial Infrared Interrupt Enable.</b> Enables/Disables the serial infrared interrupt.</p> <p>0 Serial infrared Interrupt disabled  1 Serial infrared Interrupt enabled</p>
7 WKEN	<p><b>WAKE Interrupt Enable.</b> Enables/Disables the WAKE bit to generate an interrupt. The WAKE bit is set at the detection of a start bit by the receiver.</p> <p>0 Disable the WAKE interrupt  1 Enable the WAKE interrupt</p>
6 IDDMAEN	<p><b>DMA IDLE Condition Detected Interrupt Enable</b> Enables/Disables the receive DMA request <i>dma_req_rx</i> for the IDLE interrupt (triggered with IDLE flag in USR2[12]).</p> <p>0 DMA IDLE interrupt disabled  1 DMA IDLE interrupt enabled</p>
5 IRSC	<p><b>IR Special Case.</b> Selects the clock for the vote logic. When set, IRSC switches the vote logic clock from the sampling clock to the UART reference clock. The IR pulses are counted a predetermined amount of time depending on the reference frequency. See <a href="#">InfraRed Special Case (IRSC) Bit</a>.</p> <p>0 The vote logic uses the sampling clock (16x baud rate) for normal operation  1 The vote logic uses the UART reference clock</p>
4 LPBYP	<p><b>Low Power Bypass.</b> Allows to bypass the low power new features in UART. To use during debug phase.</p> <p>0 Low power features enabled  1 Low power features disabled</p>
3 TCEN	<p><b>TransmitComplete Interrupt Enable.</b> Enables/Disables the TXDC bit to generate an interrupt (<i>interrupt_uart</i> = 0)</p> <p><b>NOTE:</b> An interrupt will be issued as long as TCEN and TXDC are high even if the transmitter is not enabled. In general, user should enable the transmitter before enabling the TXDC interrupt.</p> <p>0 Disable TXDC interrupt  1 Enable TXDC interrupt</p>
2 BKEN	<p><b>BREAK Condition Detected Interrupt Enable.</b> Enables/Disables the BRCD bit to generate an interrupt.</p> <p>0 Disable the BRCD interrupt  1 Enable the BRCD interrupt</p>
1 OREN	<p><b>Receiver Overrun Interrupt Enable.</b> Enables/Disables the ORE bit to generate an interrupt.</p> <p>0 Disable ORE interrupt  1 Enable ORE interrupt</p>

Table continues on the next page...

## UARTx\_UCR4 field descriptions (continued)

Field	Description
0 DREN	<b>Receive Data Ready Interrupt Enable.</b> Enables/Disables the RDR bit to generate an interrupt.  0    Disable RDR interrupt 1    Enable RDR interrupt

## 51.15.8 UART FIFO Control Register (UARTx\_UFCR)

Address: Base address + 90h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TXTL							RFDIV		DCEDTE	RXTL					
W																
Reset	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1

## UARTx\_UFCR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–10 TXTL	<b>Transmitter Trigger Level.</b> Controls the threshold at which a maskable interrupt is generated by the TxFIFO. A maskable interrupt is generated whenever the data level in the TxFIFO falls below the selected threshold. The bits are encoded as shown in the Settings column.  Settings 0 to 32 are in use. All other settings are Reserved.  000000    Reserved 000001    Reserved 000010    TxFIFO has 2 or fewer characters ...        — ...        — 011111    TxFIFO has 31 or fewer characters 100000    TxFIFO has 32 characters (maximum)
9–7 RFDIV	<b>Reference Frequency Divider.</b> Controls the divide ratio for the reference clock. The input clock is <i>module_clock</i> . The output from the divider is <i>ref_clk</i> which is used by BRM to create the 16x baud rate oversampling clock ( <i>brm_clk</i> ).  000    Divide input clock by 6 001    Divide input clock by 5

Table continues on the next page...



**UARTx\_UFCR field descriptions (continued)**

Field	Description
	010 Divide input clock by 4 011 Divide input clock by 3 100 Divide input clock by 2 101 Divide input clock by 1 110 Divide input clock by 7 111 Reserved
6 DCEDTE	<b>DCE/DTE mode select.</b> Select UART as data communication equipment (DCE mode) or as data terminal equipment (DTE mode).  0 DCE mode selected 1 DTE mode selected
5–0 RXTL	<b>Receiver Trigger Level.</b> Controls the threshold at which a maskable interrupt is generated by the RxFIFO. A maskable interrupt is generated whenever the data level in the RxFIFO reaches the selected threshold. The RXTL bits are encoded as shown in the Settings column.  Setting 0 to 32 are in use. All other settings are Reserved.  000000 0 characters received 000001 RxFIFO has 1 character ... — ... — 011111 RxFIFO has 31 characters 100000 RxFIFO has 32 characters (maximum)

## 51.15.9 UART Status Register 1 (UARTx\_USR1)

Address: Base address + 94h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	PARITYERR	RTSS	TRDY	RTSD	ESCF	FRAMERR	RRDY	AGTIM	DTRD	RXDS	AIRINT	AWAKE	SAD	Reserved		
W	w1c			w1c	w1c	w1c		w1c	w1c		w1c	w1c	w1c			
Reset	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0

**UARTx\_USR1 field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15 PARITYERR	<b>Parity Error Interrupt Flag.</b> Indicates a parity error is detected. PARITYERR is cleared by writing 1 to it. Writing 0 to PARITYERR has no effect. When parity is disabled, PARITYERR always reads 0. At reset, PARITYERR is set to 0.  0 No parity error detected 1 Parity error detected (write 1 to clear)
14 RTSS	<b>RTS_B Pin Status.</b> Indicates the current status of the RTS_B pin. A "snapshot" of the pin is taken immediately before RTSS is presented to the data bus. RTSS cannot be cleared because all writes to RTSS are ignored. At reset, RTSS is set to 0.

*Table continues on the next page...*

## UARTx\_USR1 field descriptions (continued)

Field	Description
	0 The RTS_B pin is high (inactive) 1 The RTS_B pin is low (active)
13 TRDY	<b>Transmitter Ready Interrupt / DMA Flag.</b> Indicates that the TxFIFO emptied below its target threshold and requires data. TRDY is automatically cleared when the data level in the TxFIFO exceeds the threshold set by TXTL bits. At reset, TRDY is set to 1.  0 The transmitter does not require data 1 The transmitter requires data (interrupt posted)
12 RTSD	<b>RTS Delta.</b> Indicates whether the RTS_B pin changed state. It (RTSD) generates a maskable interrupt. When in STOP mode, RTS assertion sets RTSD and can be used to wake the processor. The current state of the RTS_B pin is available on the RTSS bit. Clear RTSD by writing 1 to it. Writing 0 to RTSD has no effect. At reset, RTSD is set to 0.  0 RTS_B pin did not change state since last cleared 1 RTS_B pin changed state (write 1 to clear)
11 ESCF	<b>Escape Sequence Interrupt Flag.</b> Indicates if an escape sequence was detected. ESCF is asserted when the ESCEN bit is set and an escape sequence is detected in the RxFIFO. Clear ESCF by writing 1 to it. Writing 0 to ESCF has no effect.  0 No escape sequence detected 1 Escape sequence detected (write 1 to clear).
10 FRAMERR	<b>Frame Error Interrupt Flag.</b> Indicates that a frame error is detected. The <i>interrupt_uart</i> interrupt will be generated if a frame error is detected and the interrupt is enabled. Clear FRAMERR by writing 1 to it. Writing 0 to FRAMERR has no effect.  0 No frame error detected 1 Frame error detected (write 1 to clear)
9 RRDY	<b>Receiver Ready Interrupt / DMA Flag.</b> Indicates that the RxFIFO data level is above the threshold set by the RXTL bits. (See the RXTL bits description in <a href="#">UART FIFO Control Register (UART_UFCR)</a> for setting the interrupt threshold.) When asserted, RRDY generates a maskable interrupt or DMA request. RRDY is automatically cleared when data level in the RxFIFO goes below the set threshold level. At reset, RRDY is set to 0.  0 No character ready 1 Character(s) ready (interrupt posted)
8 AGTIM	<b>Ageing Timer Interrupt Flag.</b> Indicates that data in the RxFIFO has been idle for a time of 8 character lengths (where a character length consists of 7 or 8 bits, depending on the setting of the WS bit in UCR2, with the bit time corresponding to the baud rate setting) and FIFO data level is less than RxFIFO threshold level (RXTL in the UFCR). Clear by writing a 1 to it.  0 AGTIM is not active 1 AGTIM is active (write 1 to clear)
7 DTRD	<b>DTR Delta.</b> Indicates whether DTR_B (in DCE mode) or DSR_B (in DTE mode) pins changed state. DTRD generates a maskable interrupt if DTRDEN (UCR3[3]) is set. Clear DTRD by writing 1 to it. Writing 0 to DTRD has no effect.  0 DTR_B (DCE) or DSR_B (DTE) pin did not change state since last cleared 1 DTR_B (DCE) or DSR_B (DTE) pin changed state (write 1 to clear)
6 RXDS	<b>Receiver IDLE Interrupt Flag.</b> Indicates that the receiver state machine is in an IDLE state, the next state is IDLE, and the receive pin is high. RXDS is automatically cleared when a character is received. RXDS is active only when the receiver is enabled.

Table continues on the next page...

## UARTx\_USR1 field descriptions (continued)

Field	Description
	0 Receive in progress 1 Receiver is IDLE
5 AIRINT	Asynchronous IR WAKE Interrupt Flag. Indicates that the IR WAKE pulse was detected on the RXD pin. Clear AIRINT by writing 1 to it. Writing 0 to AIRINT has no effect.  0 No pulse was detected on the RXD IrDA pin 1 A pulse was detected on the RXD IrDA pin
4 AWAKE	Asynchronous WAKE Interrupt Flag. Indicates that a falling edge was detected on the RXD pin. Clear AWAKE by writing 1 to it. Writing 0 to AWAKE has no effect.  0 No falling edge was detected on the RXD Serial pin 1 A falling edge was detected on the RXD Serial pin
3 SAD	RS-485 Slave Address Detected Interrupt Flag.  Indicates if RS-485 Slave Address was detected . SAD was asserted in RS-485 mode when the SADEN bit is set and Slave Address is detected in RxFIFO (in Nomal Address Detect Mode, the 9 <sup>th</sup> data bit = 1; in Automatic Address Detect Mode, the received charater matches the programmed SLADDR).  0 No slave address detected 1 Slave address detected
2-0 -	This field is reserved. Reserved

## 51.15.10 UART Status Register 2 (UARTx\_USR2)

Address: Base address + 98h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ADET	TXFE	DTRF	IDLE	ACST	RIDELT	RIIN	IRINT	WAKE	DCDDELT	DCDIN	RTSF	TXDC	BRCD	ORE	RDR
W	w1c		w1c	w1c	w1c	w1c		w1c	w1c	w1c		w1c		w1c	w1c	
Reset	0	1	0	0	0	0	0	0	0	0	1	0	1	0	0	0

**UARTx\_USR2 field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15 ADET	<b>Automatic Baud Rate Detect Complete.</b> Indicates that an "A" or "a" was received and that the receiver detected and verified the incoming baud rate. Clear ADET by writing 1 to it. Writing 0 to ADET has no effect.  0 ASCII "A" or "a" was not received 1 ASCII "A" or "a" was received (write 1 to clear)
14 TXFE	<b>Transmit Buffer FIFO Empty.</b> Indicates that the transmit buffer (TxFIFO) is empty. TXFE is cleared automatically when data is written to the TxFIFO. Even though TXFE is high, the transmission might still be in progress.  0 The transmit buffer (TxFIFO) is not empty 1 The transmit buffer (TxFIFO) is empty

*Table continues on the next page...*

## UARTx\_USR2 field descriptions (continued)

Field	Description
13 DTRF	<p><b>DTR edge triggered interrupt flag.</b> This bit is asserted, when the programmed edge is detected on the DTR_B pin (DCE mode) or on DSR_B (DTE mode). This flag can cause an interrupt if DTREN (UCR3[13]) is enabled.</p> <p>0 Programmed edge not detected on DTR/DSR 1 Programmed edge detected on DTR/DSR (write 1 to clear)</p>
12 IDLE	<p><b>Idle Condition.</b> Indicates that an idle condition has existed for more than a programmed amount frame (see <a href="#">Idle Line Detect</a>). An interrupt can be generated by this IDLE bit if IDEN (UCR1[12]) is enabled. IDLE is cleared by writing 1 to it. Writing 0 to IDLE has no effect.</p> <p>0 No idle condition detected 1 Idle condition detected (write 1 to clear)</p>
11 ACST	<p><b>Autobaud Counter Stopped.</b> In autobaud detection (ADBR=1), indicates the counter which determines the baud rate was running and is now stopped. This means either START bit is finished (if ADNIMP=1), or Bit 0 is finished (if ADNIMP=0). See <a href="#">New Autobaud Counter Stopped bit and Interrupt</a>, for more details. An interrupt can be flagged on <i>interrupt_uart</i> if ACIEN=1.</p> <p>0 Measurement of bit length not finished (in autobaud) 1 Measurement of bit length finished (in autobaud). (write 1 to clear)</p>
10 RIDEIT	<p><b>Ring Indicator Delta.</b> This bit is used in DTE mode to indicate that the Ring Indicator input (RI_B) has changed state. This flag can generate an interrupt if RI (UCR3[8]) is enabled. RIDEIT is cleared by writing 1 to it. Writing 0 to RIDEIT has no effect.</p> <p>0 Ring Indicator input has not changed state 1 Ring Indicator input has changed state (write 1 to clear)</p>
9 RIIN	<p><b>Ring Indicator Input.</b> This bit is used in DTE mode to reflect the status if the Ring Indicator input (RI_B). The Ring Indicator input is used to indicate that a ring has occurred. In DCE mode this bit is always zero.</p> <p>0 Ring Detected 1 No Ring Detected</p>
8 IRINT	<p><b>Serial Infrared Interrupt Flag.</b> When an edge is detected on the RXD pin during SIR Mode, this flag will be asserted. This flag can cause an interrupt which can be masked using the control bit ENIRI: UCR4 [8].</p> <p>0 no edge detected 1 valid edge detected (write 1 to clear)</p>
7 WAKE	<p><b>Wake.</b> Indicates the start bit is detected. WAKE can generate an interrupt that can be masked using the WKEN bit. Clear WAKE by writing 1 to it. Writing 0 to WAKE has no effect.</p> <p>0 start bit not detected 1 start bit detected (write 1 to clear)</p>
6 DCDDELT	<p><b>Data Carrier Detect Delta.</b> This bit is used in DTE mode to indicate that the Data Carrier Detect input (DCD_B) has changed state.</p> <p>This flag can cause an interrupt if DCD (UCR3[9]) is enabled. When in STOP mode, this bit can be used to wake the processor. In DCE mode this bit is always zero.</p> <p>0 Data Carrier Detect input has not changed state 1 Data Carrier Detect input has changed state (write 1 to clear)</p>
5 DCDIN	<p><b>Data Carrier Detect Input.</b> This bit is used in DTE mode reflect the status of the Data Carrier Detect input (DCD_B). The Data Carrier Detect input is used to indicate that a carrier signal has been detected. In DCE mode this bit is always zero.</p>

Table continues on the next page...

## UARTx\_USR2 field descriptions (continued)

Field	Description
	0 Carrier signal Detected 1 No Carrier signal Detected
4 RTSF	<b>RTS Edge Triggered Interrupt Flag.</b> Indicates if a programmed edge is detected on the RTS_B pin. The RTEC bits select the edge that generates an interrupt (see <a href="#">Table 51-8</a> ). RTSF can generate an interrupt that can be masked using the RTSEN bit. Clear RTSF by writing 1 to it. Writing 0 to RTSF has no effect.  0 Programmed edge not detected on RTS_B 1 Programmed edge detected on RTS_B (write 1 to clear)
3 TXDC	<b>Transmitter Complete.</b> Indicates that the transmit buffer (TxFIFO) and Shift Register is empty; therefore the transmission is complete. TXDC is cleared automatically when data is written to the TxFIFO.  0 Transmit is incomplete 1 Transmit is complete
2 BRCD	<b>BREAK Condition Detected.</b> Indicates that a BREAK condition was detected by the receiver. Clear BRCD by writing 1 to it. Writing 0 to BRCD has no effect.  0 No BREAK condition was detected 1 A BREAK condition was detected (write 1 to clear)
1 ORE	<b>Overrun Error.</b> When set to 1, ORE indicates that the receive buffer (Rx FIFO) was full (32 chars inside), and a 33rd character has been fully received. This 33rd character has been discarded. Clear ORE by writing 1 to it. Writing 0 to ORE has no effect.  0 No overrun error 1 Overrun error (write 1 to clear)
0 RDR	<b>Receive Data Ready</b> -Indicates that at least 1 character is received and written to the Rx FIFO. If the URXD register is read and there is only 1 character in the Rx FIFO, RDR is automatically cleared.  0 No receive data ready 1 Receive data ready

## 51.15.11 UART Escape Character Register (UARTx\_UESC)

Address: Base address + 9Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																ESC_CHAR															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1	1

## UARTx\_UESC field descriptions

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value 0.
7–0 ESC_CHAR	<b>UART Escape Character.</b> Holds the selected escape character that all received characters are compared against to detect an escape sequence.

## 51.15.12 UART Escape Timer Register (UARTx\_UTIM)

Address: Base address + A0h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																TIM															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### UARTx\_UTIM field descriptions

Field	Description
31–12 Reserved	This read-only field is reserved and always has the value 0.
11–0 TIM	UART Escape Timer. Holds the maximum time interval (in ms) allowed between escape characters. The escape timer register is programmable in intervals of 2 ms. See <a href="#">Escape Sequence Detection</a> and <a href="#">Table 51-13</a> for more information on the UART escape sequence detection.  Reset value 0x000 = 2 ms up to 0xFFFF = 8.192 s.

## 51.15.13 UART BRM Incremental Register (UARTx\_UBIR)

This register can be written by both software and hardware. When enabling the automatic baud rate detection feature hardware can write 0x000F value into the UBIR after finishing detecting baud rate. Hardware has higher priority when both software and hardware try to write it at the same cycle.

Please note software reset will reset the register to its reset value.

Address: Base address + A4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																INC															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### UARTx\_UBIR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–0 INC	Incremental Numerator. Holds the numerator value minus one of the BRM ratio (see <a href="#">Binary Rate Multiplier (BRM)</a> ). The UBIR register MUST be updated before the UBMR register for the baud rate to be updated correctly. If only one register is written to by software, the BRM will ignore this data until the other register is written to by software. Updating this field using byte accesses is not recommended and is undefined.

- Note: The write priority in the new design is not same as the original UART. In the original design, software has higher priority than hardware when writing this register at the same time.



### 51.15.14 UART BRM Modulator Register (UARTx\_UBMR)

This register can be written by both software and hardware. When enabling the automatic baud rate detection feature hardware can write a proper value into the UBMR based on detected baud rate. Hardware has higher priority when both software and hardware try to write it at the same cycle.

Please note software reset will reset the register to its reset value.

Address: Base address + A8h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																MOD															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### UARTx\_UBMR field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–0 MOD	<b>Modulator Denominator.</b> Holds the value of the denominator minus one of the BRM ratio (see <a href="#">Binary Rate Multiplier (BRM)</a> ). The UBIR register MUST be updated before the UBMR register for the baud rate to be updated correctly. If only one register is written to by software, the BRM will ignore this data until the other register is written to by software. Updating this register using byte accesses is not recommended and undefined.

- Note: The write priority in the new design is not same as the original UART. In the original design, software has higher priority than hardware when writing this register at the same time.

### 51.15.15 UART Baud Rate Count Register (UARTx\_UBRC)

Address: Base address + ACh offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0																BCNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

#### UARTx\_UBRC field descriptions

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–0 BCNT	<b>Baud Rate Count Register.</b> This read only register is used to count the start bit of the incoming baud rate (if ADNIMP=1), or start bit + bit0 (if ADNIMP=0). When the measurement is done, the Baud Rate Count Register contains the number of UART internal clock cycles (clock after divider) present in an incoming bit. BCNT retains its value until the next Automatic Baud Rate Detection sequence has been initiated. The 16 bit Baud Rate Count register is reset to 4 and stays at hex FFFF in the case of an overflow.

51.15.16 UART One Millisecond Register (UARTx\_ONEMS)

NOTE

This register has been expanded from 16 bits to 24 bits. In previous versions, the 16-bit ONEMS can only support the maximum 65.535MHz (0xFFFFx1000) *ref\_clk*. To support 4Mbps Bluetooth application with 66.5MHz *module\_clock*, the value 0x103C4 (66.5M/1000) should be written into this register. In this case, the 16 bits are not enough to contain the 0x103C4. So this register was expanded to 24 bits to support high frequency of the *ref\_clk*.

Address: Base address + B0h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								ONEMS																							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

UARTx\_ONEMS field descriptions

Field	Description
31–24 Reserved	This read-only field is reserved and always has the value 0.
23–0 ONEMS	<p><b>One Millisecond Register.</b> This 24-bit register must contain the value of the UART internal frequency (<i>ref_clk</i> in <a href="#">Figure 51-1</a>) divided by 1000. The internal frequency is obtained after the UART BRM internal divider (<math>F (ref\_clk) = F(module\_clock) / RFDIV</math>).</p> <p>In fact this register contains the value corresponding to the number of UART BRM internal clock cycles present in one millisecond.</p> <p>The ONEMS (and UTIM) registers value are used in the escape character detection feature (<a href="#">Escape Sequence Detection</a>) to count the number of clock cycles left between two escape characters. The ONEMS register is also used in infrared special case mode (IRSC = UCR4[5] = 1'b1), see <a href="#">InfraRed Special Case (IRSC) Bit</a>.</p>

## 51.15.17 UART Test Register (UARTx\_UTS)

Address: Base address + B4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0		FRCPERR	LOOP	DBGEN	LOOPIR	RXDBG	0	TXEMPTY	RXEMPTY	TXFULL	RXFULL	0			
W																
Reset	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0

### UARTx\_UTS field descriptions

Field	Description
31–14 Reserved	This read-only field is reserved and always has the value 0.
13 FRCPERR	Force Parity Error. Forces the transmitter to generate a parity error if parity is enabled. FRCPERR is provided for system debugging.  0 Generate normal parity 1 Generate inverted parity (error)
12 LOOP	Loop TX and RX for Test. Controls loopback for test purposes. When LOOP is high, the receiver input is internally connected to the transmitter and ignores the RXD pin. The transmitter is unaffected by LOOP. If RXDMUXSEL (UCR3[2]) is set to 1, the loopback is applied on serial and IrDA signals. If RXDMUXSEL is set to 0, the loopback is only applied on serial signals.  0 Normal receiver operation 1 Internally connect the transmitter output to the receiver input
11 DBGEN	debug_enable_B. This bit controls whether to respond to the <i>debug_req</i> input signal.  0 UART will go into debug mode when <i>debug_req</i> is HIGH 1 UART will not go into debug mode even if <i>debug_req</i> is HIGH
10 LOOPIR	<b>Loop TX and RX for IR Test (LOOPIR).</b> This bit controls loopback from transmitter to receiver in the InfraRed interface.  0 No IR loop 1 Connect IR transmitter to IR receiver
9 RXDBG	<b>RX_fifo_debug_mode.</b> This bit controls the operation of the RX fifo read counter when in debug mode.  0 rx fifo read pointer does not increment 1 rx_fifo read pointer increments as normal

Table continues on the next page...

## UARTx\_UTS field descriptions (continued)

Field	Description
8–7 Reserved	This read-only field is reserved and always has the value 0.
6 TXEMPTY	TxFIFO Empty. Indicates that the TxFIFO is empty. 0 The TxFIFO is not empty 1 The TxFIFO is empty
5 RXEMPTY	RxFIFO Empty. Indicates the RxFIFO is empty. 0 The RxFIFO is not empty 1 The RxFIFO is empty
4 TXFULL	TxFIFO FULL. Indicates the TxFIFO is full. 0 The TxFIFO is not full 1 The TxFIFO is full
3 RXFULL	RxFIFO FULL. Indicates the RxFIFO is full. 0 The RxFIFO is not full 1 The RxFIFO is full
2–1 Reserved	This read-only field is reserved and always has the value 0.
0 SOFTTRST	Software Reset. Indicates the status of the software reset (SRST_B bit of UCR2). 0 Software reset inactive 1 Software reset active

## 51.15.18 UART RS-485 Mode Control Register (UARTx\_UMCR)

Address: Base address + B8h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SLADDR								0				SADEN	TXB8	SLAM	MDEN
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**UARTx\_UMCR field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–8 SLADDR	RS-485 Slave Address Character. Holds the selected slave address character that the receiver will try to detect.
7–4 Reserved	This read-only field is reserved and always has the value 0.
3 SADEN	RS-485 Slave Address Detected Interrupt Enable. 0 Disable RS-485 Slave Address Detected Interrupt 1 Enable RS-485 Slave Address Detected Interrupt
2 TXB8	Transmit RS-485 bit 8 (the ninth bit or 9 <sup>th</sup> bit). In RS-485 mode, software writes TXB8 bit as the 9 <sup>th</sup> data bit to be transmitted. 0 0 will be transmitted as the RS485 9 <sup>th</sup> data bit 1 1 will be transmitted as the RS485 9 <sup>th</sup> data bit
1 SLAM	RS-485 Slave Address Detect Mode Selection. 0 Select Normal Address Detect mode 1 Select Automatic Address Detect mode
0 MDEN	9-bit data or Multidrop Mode (RS-485) Enable. 0 Normal RS-232 or IrDA mode, see <a href="#">Table 51-1</a> for detail. 1 Enable RS-485 mode, see <a href="#">Table 51-1</a> for detail



## Chapter 52

# Universal Serial Bus Controller (USB)

### 52.1 Overview

The USB controller block provides high performance USB functionality that conforms to the *Universal Serial Bus Specification*, Rev. 2.0 (Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, Philips; 2000), and the *On-The-Go and Embedded Host Supplement to the USB Revision 2.0 Specification* (Hewlett-Packard Company, Intel Corporation, LSI Corporation, Microsoft Corporation, Renesas Electronics Corporation, ST-Ericsson; 2012). See [Features](#) for more details. All four controller cores are single-port cores. For the OTG core, there is only one port. It can be used as either a downstream or an upstream port. For the host-only core, there is also only one port which is used as a downstream port.

The USB controller consists of three independent USB controller cores: two On-The-Go (OTG) controller cores, and one host-only controller core. Each controller core can support UTMI or HSIC interfaces according to its feature. See [Features](#) for more details. All three controller cores are single-port cores. For the OTG core, there is only one port. It can be used as either a downstream or an upstream port. For the host-only core, there is also only one port which is used as a downstream port.

The following figure is a block diagram of USB.

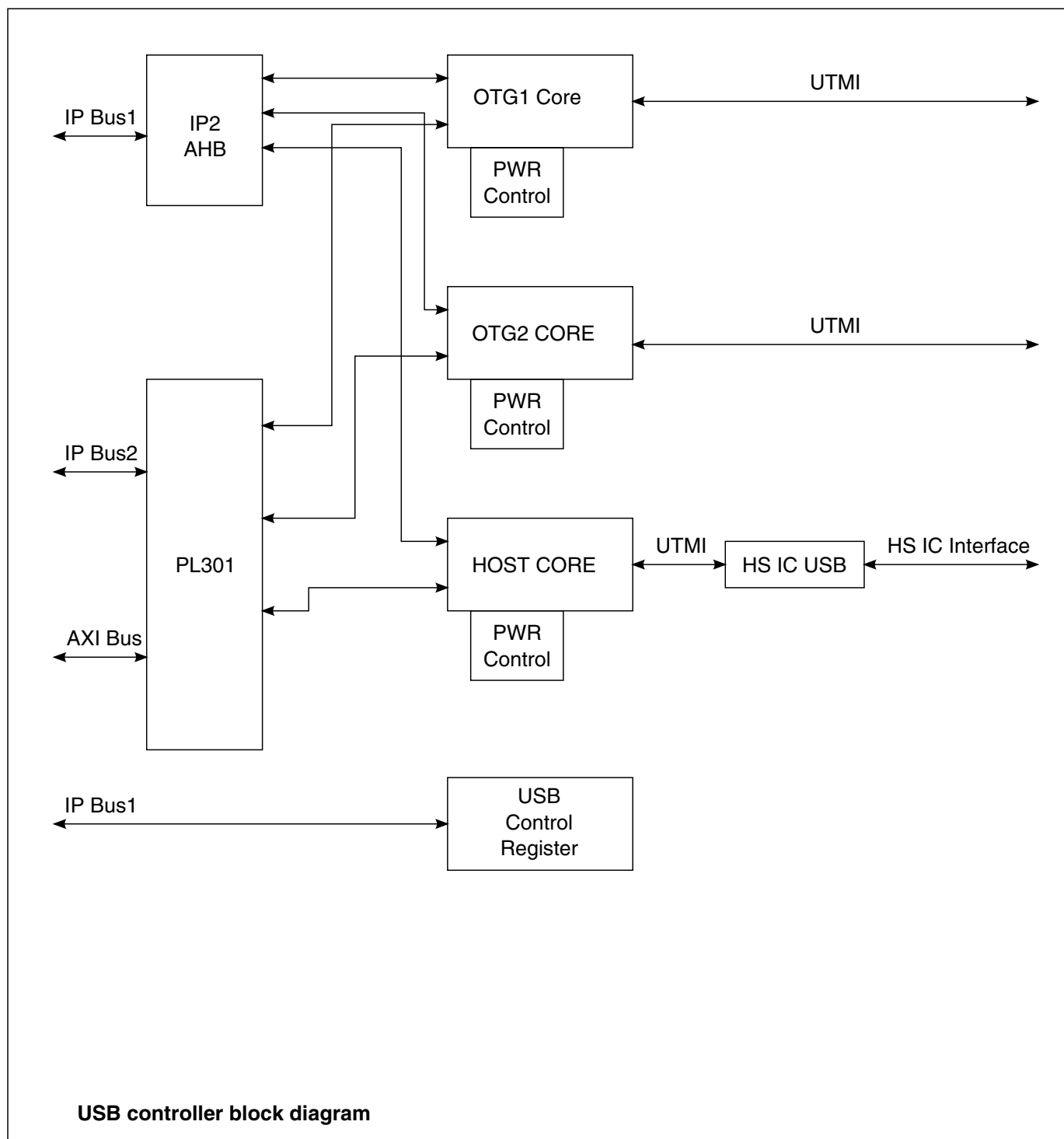


Figure 52-1. USB block diagram



## 52.1.1 Features

There are three USB 2.0 controller cores in this chip:

- Controller Core 0 is also named 'OTG1 Core'; its connected port is named 'OTG1 port'.
- Controller Core 1 is also named 'OTG2 Core'; its connected port is named 'OTG2 port'.
- Controller Core 2 is also named 'Host1 Core'; its connected port is named 'Host1 port'.

The following list provides features of each of the controller cores.

- USB 2.0 Controller Core 0
  - High-Speed/Full-Speed/Low-Speed OTG core
  - HS/FS/LS UTMI compliant interface
  - High Speed, Full Speed and Low Speed operation in Host mode (with UTMI transceiver)
  - High Speed, and Full Speed operation in Peripheral mode (with UTMI transceiver)
  - Hardware support for OTG signaling, session request protocol, and host negotiation protocol
  - Up to 8 bidirectional endpoints
  - Support charger detection
- USB 2.0 Controller Core 1
  - High-Speed/Full-Speed/Low-Speed OTG core
  - HS/FS/LS UTMI compliant interface
  - High Speed, Full Speed and Low Speed operation in Host mode (with UTMI transceiver)
  - High Speed, and Full Speed operation in Peripheral mode (with UTMI transceiver)
  - Hardware support for OTG signaling, session request protocol, and host negotiation protocol
  - Up to 8 bidirectional endpoints
- USB 2.0 Controller Core 2
  - High-Speed/Full-Speed/Low-Speed Host-Only core
  - High Speed Inter-Chip USB compliant interface (HSIC)
- Low-power mode with local and remote wake-up capability
- Serial PHY interfaces configurable for bidirectional/unidirectional and differential/single ended
- Embedded DMA controller in each core

## 52.1.2 Modes of Operation

The USB has two main modes of operation: normal mode and low power mode.

Each USB controller core can operate in High Speed operation (480 Mbps), Full Speed operation (12Mbps) and Low Speed operation (1.5 Mbps).

This chapter explains the operation modes.

### 52.1.2.1 Normal Mode

The OTG controller core can operate in Host mode and Device (Peripheral) mode. The host-only controller core can operate in Host mode only.

Each USB controller core has its corresponding port, which can work in one or more interface modes.

#### NOTE

Each controller supports only the interface type listed below. Selecting a different interface type in the PORTSC.PTS field results in unpredictable behavior and may cause the system to hang.

- OTG1 port
  - This port supports on-chip UTMI transceiver only.
- OTG2 port
  - This port supports on-chip UTMI transceiver only.
- Host1 Port
  - This port supports HSIC interface only.

Interface for on-board HSIC compatible USB peripherals.

#### NOTE

HSIC is an inter-chip interface that is optimized for circuit board layouts.

### 52.1.2.2 Low-Power Mode

Each USB controller core has a low-power mode (Suspend mode) to save power consumption.

As described in the USB 2.0 specification, the device can go into the Suspend state after it sees a constant Idle state on the upstream facing port. The OTG controller core enters Suspend mode after 3 ms of inactivity on the port when it is in Device Operation mode. Host controllers, including the OTG controller in Host mode, do not suspend automatically but can be placed in Suspend mode by software.

Either the local ARM platform or the remote USB Host/Peripheral can initiate a wake-up sequence to resume USB communication. For details about Suspend/Resume, see [USB Power Control](#).

## 52.2 External Signals

The table found here describes the external signals of USB.

**Table 52-1. USB External Signals**

Signal	Description	Pad	Mode	Direction
USB_H_DATA	Data signal	HSIC_DAT	ALT0	IO
USB_H_STROBE	Strobe signal	HSIC_STROBE	ALT0	IO
USB_OTG1_DN	DN OTG1 Signal	USB_OTG1_DN	No muxing	IO
USB_OTG1_DP	DP OTG1 Signal	USB_OTG1_DP	No muxing	IO
USB_OTG1_ID	ID signal	EPDC_PWRCOM	ALT4	I
		FEC_RXD0	ALT2	
		LCD_DAT1	ALT2	
		REF_CLK_32K	ALT3	
		SD3_DAT0	ALT4	
USB_OTG1_OC	OTG1 External input for VBUS overcurrent detection	ECSPi2_MISO	ALT6	I
		KEY_ROW4	ALT6	
		SD3_DAT3	ALT6	
USB_OTG1_PWR	To control PMIC to supply VBUS voltage	ECSPi2_SS0	ALT6	O
		KEY_COL4	ALT6	
		SD3_CLK	ALT6	
USB_OTG2_DN	DN OTG2 Signal	USB_OTG2_DN	No muxing	IO
USB_OTG2_DP	DP OTG2 Signal	USB_OTG2_DP	No muxing	IO
USB_OTG2_ID	ID signal	EPDC_PWRINT	ALT4	I
		LCD_DAT0	ALT2	
		REF_CLK_24M	ALT3	
		SD3_CMD	ALT4	

*Table continues on the next page...*

**Table 52-1. USB External Signals  
(continued)**

Signal	Description	Pad	Mode	Direction
USB_OTG2_OC	OTG2 External input for VBUS overcurrent detection	ECSP11_SCLK	ALT6	I
		ECSP12_SCLK	ALT6	
		KEY_ROW5	ALT6	
		SD3_DAT2	ALT6	
USB_OTG2_PWR	To control PMIC to supply VBUS voltage	ECSP11_SS0	ALT6	O
		KEY_COL5	ALT6	
		SD3_CMD	ALT6	
USB_OTG_CHD_B	Charge detect signal	USB_OTG_CHD_B	No muxing	IO

## 52.3 Functional Description

These sections describe the functionality of the various building blocks of the USB.

### 52.3.1 USB 2.0 Controller Core 0

The USB 2.0 Controller 0 is an instantiation of an EHCI-compatible core which supports high-, full-, and low-speed operation.

In Host mode, this controller core supports high-, full-, and low-speed operation. In Device mode, it supports high- and full-speed operation.

#### 52.3.1.1 Host Mode

The controller supports direct connection of .

Although there is no separate Transaction Translator block in the system, the transaction translator function normally associated with a USB 2.0 high speed hub has been implemented within the DMA and protocol engine blocks to support connection to full and low speed devices.

#### 52.3.1.2 Peripheral (Device) Mode

- Up to eight bidirectional endpoints
- High/full-speed operation

- Support of HNP and SRP
- Remote wake-up capability

### 52.3.1.3 Pins Used for OTG 1 Controller

The required power supplies:

- 1.1V voltage supply from LDO 1P1 regulator
- 2.5V voltage supply from LDO 2P5 regulator
- 3.0V voltage supply from USB LDO regulator

USB OTG 1 PHY pins:

- USB\_OTG1\_VBUS
- USB\_OTG1\_DN
- USB\_OTG1\_DP
- USB\_OTG1\_CHD\_B, see [Universal Serial Bus 2.0 Integrated PHY \(USB-PHY\)](#).

The following external signals are multiplexed with other pins. For the pin mapping, see [External Signals](#). For the IOMUXC register setting, see [IOMUX Controller \(IOMUXC\)](#)

- USB\_OTG1\_ID
- USB\_OTG1\_PWR
- USB\_OTG1\_OC

## 52.3.2 USB 2.0 Controller Core 1

USB 2.0 Controller Core 1 is an instantiation of EHCI-compatible core which supports High Speed / Full Speed / Low Speed operation.

### 52.3.2.1 Pins Used for OTG 2 Controller

The required power supplies:

- 1.1V voltage supply from LDO 1P1 regulator
- 2.5V voltage supply from LDO 2P5 regulator
- 3.0V voltage supply from USB LDO regulator

USB OTG 2 PHY pins:

- USB\_OTG2\_VBUS

- USB\_OTG2\_DN
- USB\_OTG2\_DP

The following external signals are multiplexed with other pins. For the pin mapping, see [External Signals](#). For the IOMUXC register setting, see [IOMUX Controller \(IOMUXC\)](#)

- USB\_OTG2\_ID
- USB\_OTG2\_PWR
- USB\_OTG2\_OC

### 52.3.3 USB 2.0 Controller Core 2

USB 2.0 Controller Core 2 is an instantiation of the EHCI-compatible core which supports High Speed / Full Speed / Low Speed operation.

This USB core's signals connect directly to I/O pins (HSIC interface).

### 52.3.4 USB Power Control

The USB controller supports suspend and wake-up functionality.

The power control block allows for placing the transceiver in USB low power mode when USB bus is IDLE, and supports local and remote wake-up to bring the transceiver out of USB low power mode when needed. Additionally, the power control block can wake-up the ARM platform from core sleep mode by generating an interrupt.

#### 52.3.4.1 Entering Low Power Suspend Mode

In Host operation mode, low power suspend mode is entered as follows:

1. Clear the ASE and PSE bits in USB\_USBCMD, and wait until the AS and PS bits in USB\_USBSTS become "0".
2. Set the "SUSPEND" bit in USB\_PORTSC1.
3. Set the "PHCD" bit in USB\_PORTSC1.
4. Set all PWD bits in USBPHY<sub>x</sub>\_PWD
5. Set CLKGATE in USBPHY<sub>x</sub>\_CTRL

#### **NOTE**

Step 3,4,5 shall be done in atomic operation. That is, interrupt should be disabled during these three steps.

For device operation mode, low power suspend mode is entered as follows:

1. After Host drive is IDLE for 3ms, an SLI interrupt is issued (the "DCSUSPEND" or "SLI" bit in USB\_USBSTS).
2. Set the "PHCD" bit on USB\_PORTSC1
3. Set all PWD bits in PHYPWD
4. Set CLKGATE in PHYCTRL

### **NOTE**

Step 2,3,4 shall be done in atomic operation. That is, interrupt should be disabled during these three steps.

## **52.3.4.2 Wake-Up Events**

The power control block monitors the USB bus when the USB core is in the USB suspend state.

Depending on whether the core is on Host or Device mode, a number of wake-up conditions are monitored. Upon detection of a wake-up condition, an interrupt (asynchronous) will be generated to ARM platform if the related wake-up interrupt enable bit is set.

USB wake-up interrupt also re-activates the ARM platform clocks if they were stopped during the suspend.

### **52.3.4.2.1 Host Mode Events**

The host controller wakes up on the following events:

- Remote Wake-up Request

A peripheral can request the host to reactivate the bus by driving wake-up signaling on the DM/DP lines. The power control block sends a wake-up request to the USB core when a J-K transition on DM/DP line is detected.

- Wake-Up On Overcurrent

If Wake-Up On Overcurrent is enabled (WKOC bit in the USB core register PORTSC1 is set '1'), the power control block sends a wake-up request to the USB core when an overcurrent event is detected.

- Wake-Up On Disconnect

If Wake-Up On Disconnect is enabled (WKDC bit in the USB core register PORTSC1 is set '1'), the power control block sends a wake-up request to the USB core when a disconnection event is detected (J-SE0/K-SE0 transition on DM/DP line).

- Wake-Up On Connect

If a Wake-Up On Connect is enabled (WKCN bit in the USB core register PORTSC1 is set '1'), the power control sub-block sends a wake-up request to the USB core when the connection event is detected (SE0-J/SE0-K transition on DM/DP line).

For a detailed description of register bits WKOC, WKDC, WKCN, please see [Port Status & Control \(USBC\\_n\\_PORTSC1\)](#).

## 52.3.5 Interrupts

### 52.3.5.1 USB Core Interrupts

Each USB core uses one dedicated vector in the Interrupt Table. The vector numbers associated with each of the cores can be found in the Interrupt section.

With the exception of the wake-up interrupts, all of the interrupt sources are controlled in the USB Cores. Refer to the [Interrupt Enable Register \(USBC\\_n\\_USBINTR\)](#) for details.

### 52.3.5.2 USB Wake-Up Interrupts

Each USB Core has an associated wake-up interrupt. The wake-up interrupts are generated outside of the USB controller cores, but using the same vector as the corresponding USB controller cores interrupt.

These interrupts are generated by the Power Control blocks which run on the 32KHz standby clock. The wake-up interrupt is designed to work even when the USB and ARM platform clocks are disabled, such that a wake-up condition on the USB bus can re-activate the ARM platform clocks.

Because the wake-up interrupt is generated and cleared on a 32 KHz clock, this interrupt request responds very slowly to clear actions. For this reason, the software must disable the wake-up interrupt to clear the request flag. Disabling the interrupt masks the request instantaneously as this is clocked by the ARM platform clock. The software should then wait for at least three 32 KHz clock cycles before re-enabling this interrupt to allow sufficient time for the request flag to clear. Because this interrupt is only used during low power modes of the USB, it is sufficient to enable the wake-up interrupt just prior to entering the USB suspend mode.



## 52.4 USB Operation Model

This section describes the detailed application knowledge for Host1, OTG1 and OTG2 ports. It can be generally divided in two parts, one is for Host and the other is for Device. Host part applies to host1 port, and to OTG port when operating in Host mode. Device part only applies to all OTG ports when operating in Device mode.

### 52.4.1 Register Interface

Configuration, control and status registers are divided into three categories, identification, capability and operational registers.

#### NOTE

USB controller registers support only DWORD (32-bit) access.

- Identification registers are used to declare the slave interface presence along with the complete set of the hardware configuration parameters.
- Static, read only capability registers define the software limits, restrictions, and capabilities of the host/device controller.
- Operational registers are dynamic control or status registers that may be read only, read/write, or read/write-to-clear. The following sections define the use of these registers.

EHCI registers are listed alongside device registers to show the complementary nature of host and device control.

The following table describes the Interface register sets.

**Table 52-2. Interface Register Sets**

Offset	Register Set	Explanation
000h-07Ch	Identification Registers	Identification registers are used to declare the slave interface presence and include a table of the hardware configuration parameters.
100h-124h	Capability Registers	Capability registers specify the limits, restrictions, and capabilities of a host/device controller implementation.  These values are used as parameters to the host/device controller driver.
080h-0FCh 140h-1FCh	Operational Registers	Operational registers are used by the system software to control and monitor the operational state of the host/device controller.

#### 52.4.1.1 Configuration, Control and Status Register Set

The following table describes the Device/Host capability registers.

**NOTE**

Depending on implementation, "x" can have the following values: UOG1, UOG2, UH.

**Table 52-3. Device/Host Capability Registers**

Offset	Size (Bytes)	Mnemonic	Register Name	Device Mode	Host Mode
000h	4	USB_x_ID	Identification Register	O	O
004h	4	USB_x_HWGENERAL	General Hardware Parameters	O	O
008h	4	USB_x_HWHOST	Host Hardware Parameters	X	O
00Ch	4	USB_x_HWDEVICE	Device Hardware Parameters	O	X
010h	4	USB_x_HWTXBUF	TX Buffer Hardware Parameters	O	O
014h	4	USB_x_HWRXBUF	RX Buffer Hardware Parameters	O	O
018-07Fh		-	Reserved		
080h	4	USB_x_GPTIMER0LD	General Purpose Timer #0 Load Register	O	O
084h	4	USB_x_GPTIMER0CTRL	General Purpose Timer #0 Control Register	O	O
088h	4	USB_x_GPTIMER1LD	General Purpose Timer #1 Load Register	O	O
08Ch	4	USB_x_GPTIMER1CTRL	General Purpose Timer #1 Control Register	O	O
090h	4	USB_x_SBUSCFG	System Bus Interface Configuration Register	O	O
094-09Fh		-	Reserved		
100h	1	USB_x_CAPLENGTH	Capability Register Length	O	O
101h		-	Reserved		
102h	2	USB_x_HCIVERSION	Host Controller Interface Version Number	X	O
104h	4	USB_x_HCSPARAMS	Host Controller Structural Parameters	X	O
108h	4	USB_x_HCCPARAMS	Host Controller Capability Parameters	X	O
10C-11Fh		-	Reserved		
120h	2	USB_x_DCIVERSION	Device Controller Interface Version Number	O	X
122h	2	-	Reserved		
124h	4	USB_x_DCCPARAMS	Device Controller Capability Parameters	O	X
128-13Fh		-	Reserved		
140h	4	USB_x_USBCMD	USB Command Register	O	O
144h	4	USB_x_USBSTS	USB Status Register	O	O
148h	4	USB_x_USBINTR	USB Interrupt Enable Register	O	O
14Ch	4	USB_x_FRINDEX	USB Frame Index	O	O
150h	4	-	Reserved		
154h	4	USB_x_PERIODICLISTBASE	Frame List Base Address	X	O
		USB_x_DEVICEADDR	USB Device Address	O	X

Table continues on the next page...

**Table 52-3. Device/Host Capability Registers (continued)**

Offset	Size (Bytes)	Mnemonic	Register Name	Device Mode	Host Mode
158h	4	USB_X_ASYNCCLISTADDR	Next Asynchronous List Address	X	O
	4	USB_X_ENDPOINTLISTADDR	Address at Endpoint list in memory	O	X
15Ch	4	-	Reserved		
160h	4	USB_X_BURSTSIZE	Programmable Burst Size	O	O
164h	4	USB_X_TXFILLTUNING	Host Transmit Pre-Buffer Packet Tuning	X	O
168h	4	-	Reserved		
16Ch	4	USB_X_IC_USB	IC_USB enable and voltage negotiation	O	O
170h	4	-	Reserved		
174h	4	USB_X_ENDPTNAK	Endpoint NAK register	O	X
178h	4	USB_X_ENDPTNAKEN	Endpoint NAK Enable register	O	X
17Ch	4	-	Reserved		
180h	4	USB_X_CONFIGFLAG	Configured Flag Register	X	O
184h	4	USB_X_PORTSC1	Port Status/Control Register 1	O	O
188-1A3h		-	Reserved		
1A4h	4	USB_X_OTGSC	On-The-Go Status/Control Register (OTG only)	O	O
1A8h	4	USB_X_USBMODE	USB Controller Operating Mode	O	O
1ACh	4	USB_X_ENDPTSETUPSTAT	Endpoint Setup Status	O	X
1B0h	4	USB_X_ENDPTPRIME	Endpoint Initialization	O	X
1B4h	4	USB_X_ENDPTFLUSH	Endpoint De-Initialization	O	X
1B8h	4	USB_X_ENDPTSTATUS	Endpoint Status	O	X
1BCh	4	USB_X_ENDPTCOMPLETE	Endpoint Complete	O	X
1C0	64	USB_X_ENDPTCTRL0	Endpoint Control Register 0-7	O	X
1C4		USB_X_ENDPTCTRL1			
...		....			
1DCh		USB_X_ENDPTCTRL7			

**NOTE**

"O" means the register is available in host/device operation mode;

"X" means the register is reserved in host/device operation mode

### 52.4.1.2 Identification Registers

Identification registers are used to declare the slave interface presence and include a table of the hardware configuration parameters.

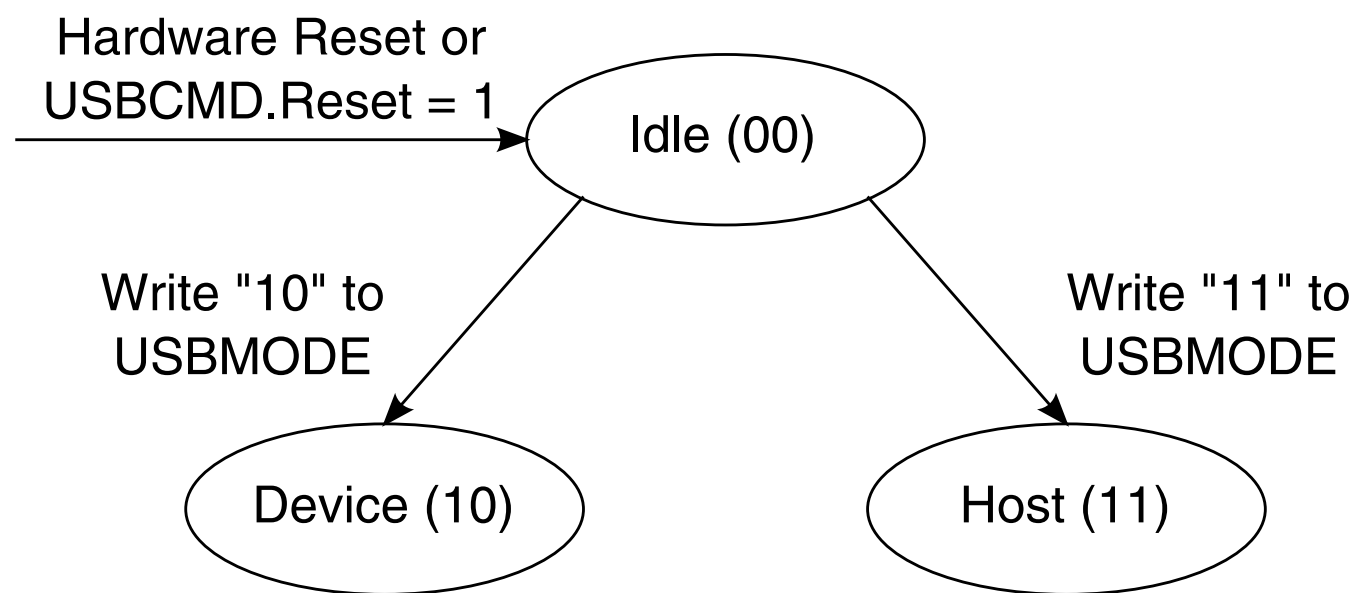
### 52.4.1.3 OTG Operations

#### 52.4.1.3.1 Register Bits

In the previous section, the Register interface has behaviors described for device mode and behaviors described for host mode. However, during OTG operations it is necessary to perform tasks independent of the controller mode.

Note also from [USB Device Mode \(USBC\\_n\\_USBMODE\)](#), that the only way to transit the controller mode out of host or device mode is with the controller reset bit. Therefore, it is also necessary for the OTG tasks to be performed independent of a controller reset as well as independent of the controller mode.

The following figure shows the controller mode.



**Figure 52-2. Controller Mode**

To this end, listed below are the register bits that are used for OTG operations, which are independent of the controller mode and are also not affected by a write to the reset bit in the USBCMD register:

All Identification Registers

All Device/Host Capability Registers

OTGSC: All bits

PORTSC1:

Physical Interface Select

Physical Interface Serial Select

Physical Interface Data Width

Physical Interface Low Power

Physical Interface Wake Signals

Port Indicators

Port Power

## 52.4.2 Host Data Structures

This section defines the interface data structures used to communicate control, status, and data between HCD (software) and the Enhanced Host Controller (hardware).

The data structure definitions in this chapter support a 32-bit memory buffer address space. The interface consists of a Periodic Schedule, Periodic Frame List, Asynchronous Schedule, Isochronous Transaction Descriptors, Split-transaction Isochronous Transfer Descriptors, Queue Heads, and Queue Element Transfer Descriptors.

The periodic frame list is the root of all periodic (isochronous and interrupt transfer type) transfers for the host controller. The asynchronous list is the root for all the bulk and control transfers. Isochronous data streams are managed using Isochronous Transaction Descriptors. Isochronous split-transaction data streams are managed with Split-transaction Isochronous Transfer Descriptors. All Interrupt, Control, and Bulk data streams are managed via queue heads and Queue Element Transfer Descriptors. These data structures are optimized to reduce the total memory footprint of the schedule and to reduce (on average) the number of memory accesses needed to execute a USB transaction.

Note that software must ensure that no interface data structure reachable by the EHCI host controller spans a 4 K-page boundary.

The data structures defined in this section are (from the host controller's perspective) a mix of read-only and read/writeable fields. The host controller must preserve the read-only fields on all data structure writes.

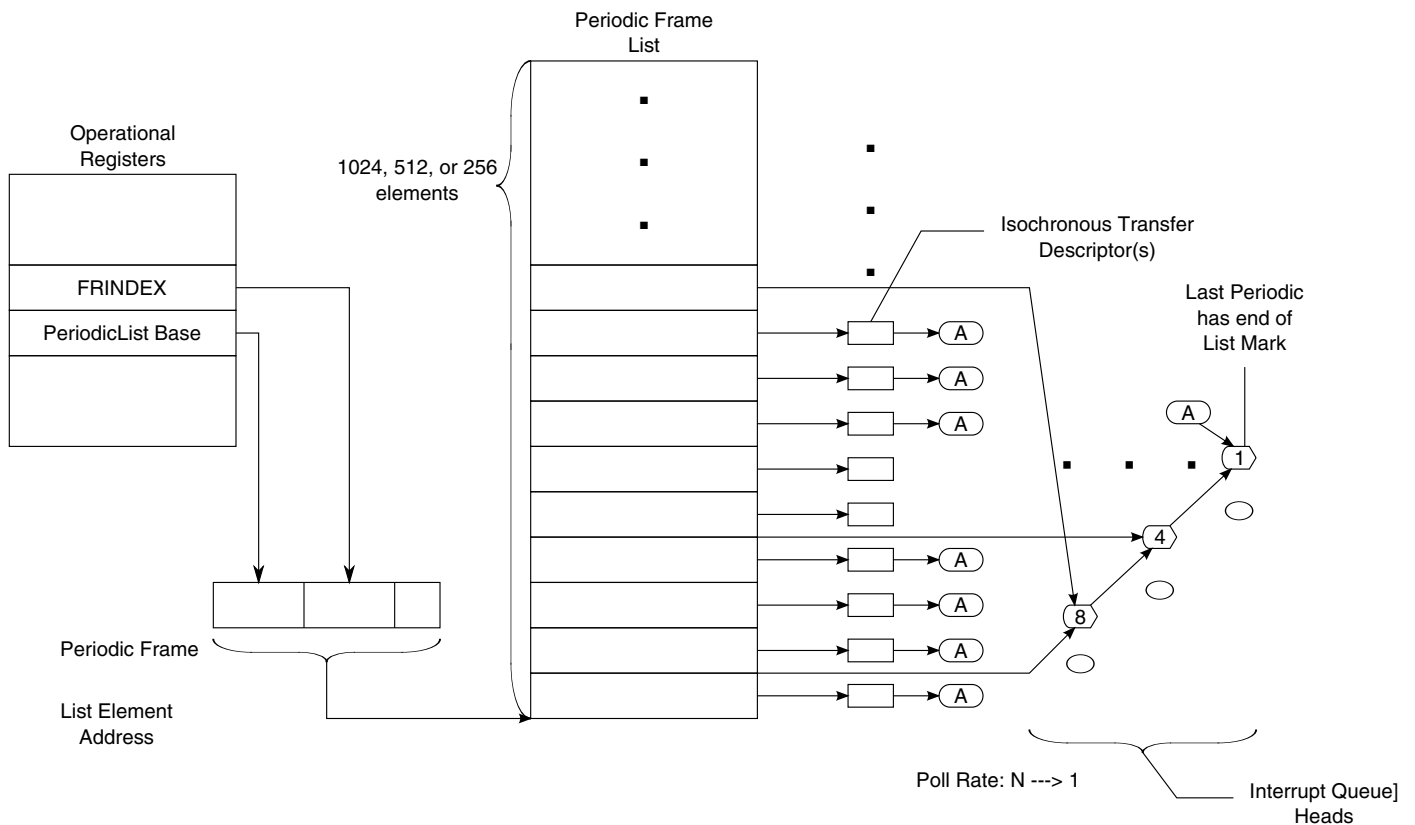
### 52.4.2.1 Periodic Frame List

This schedule is for all periodic transfers (isochronous and interrupt). The periodic schedule is referenced from the operational registers space using the USB\_PERIODICLISTBASE address register and the USB\_FRINDEX register.

The periodic schedule is based on an array of pointers called the Periodic Frame List.

The USB\_PERIODICLISTBASE address register is combined with the USB\_FRINDEX register to produce a memory pointer into the frame list. The Periodic Frame List implements a sliding window of work over time.

The following figure shows the organization of periodic schedule.



**Figure 52-3. Periodic Schedule Organization**

Split transaction Interrupt, Bulk and Control are also managed using queue heads and queue element transfer descriptors.

The periodic frame list is a 4 K-page aligned array of Frame List Link pointers. The length of the frame list may be programmable. The programmability of the periodic frame list is exported to system software via the USB\_HCCPARAMS register. If non-

programmable, the length is 1024 elements. If programmable, the length can be selected by system software as one of 256, 512, or 1024 elements. An implementation must support all three sizes. Programming the size (that is, the number of elements) is accomplished by system software writing the appropriate value into Frame List Size field in the USB\_USBCMD register.

Frame List Link pointers direct the host controller to the first work item in the frame's periodic schedule for the current micro-frame. The link pointers are aligned on DWord boundaries within the Frame List.

The table below illustrates the format of the Frame list element pointer.

**Table 52-4. Format of Frame List Element Pointer**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Frame List Link Pointer																												0	Typ			03-00H

Frame List Link pointers always reference memory objects that are 32-byte aligned. The referenced object may be an isochronous transfer descriptor for high-speed devices, a split-transaction isochronous transfer descriptor (for full-speed isochronous endpoints), or a queue head (used to support high-, full- and low-speed interrupt). System software should not place non-periodic schedule items into the periodic schedule. The least significant bits in a frame list pointer are used to key the host controller as to the type of object the pointer is referencing.

The least significant bit is the T-Bit (bit 0). When this bit is set to a one, the host controller never uses the value of the frame list pointer as a physical memory pointer. The Typ field is used to indicate the exact type of data structure being referenced by this pointer. The value encodings are.

**Table 52-5. Typ Field Value Definitions**

Value	Meaning
00b	Isochronous Transfer Descriptor
01b	Queue Head
10b	Split Transaction Isochronous Transfer Descriptor.
11b	Frame Span Traversal Node.

### 52.4.2.2 Asynchronous List Queue Head Pointer

The Asynchronous Transfer List (based at the USB\_ASYNC\_LIST\_ADDR register) is where all of the control and bulk transfers are managed.

Host controllers use this list only when it reaches the end of the periodic list, the periodic list is disabled, or the periodic list is empty.

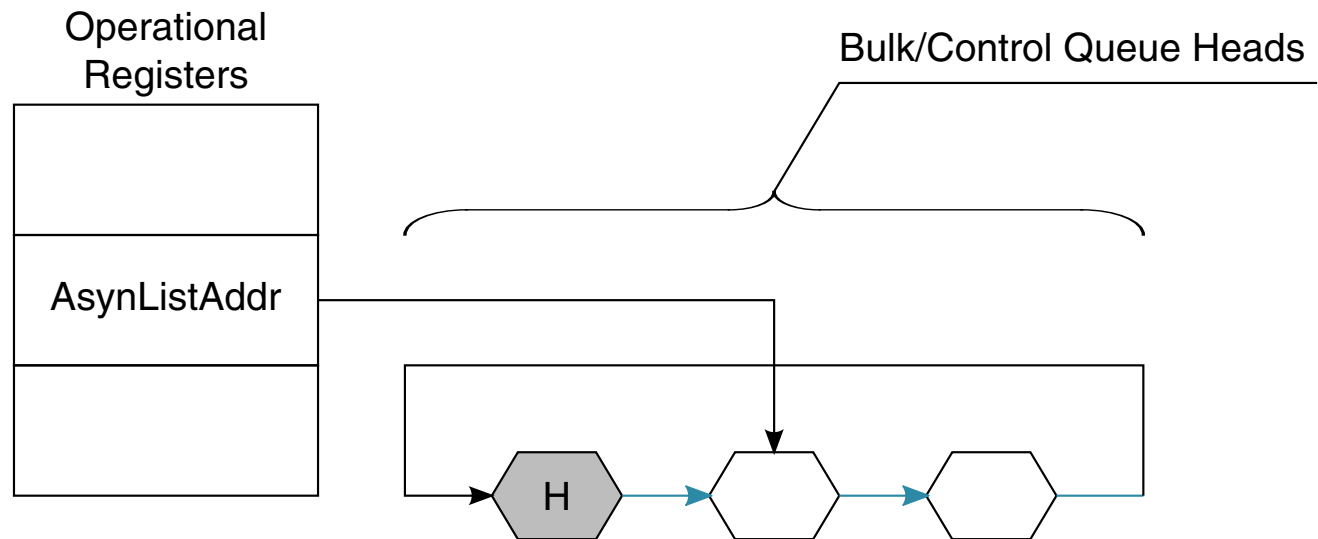


Figure 52-4. Asynchronous Schedule Organization

The Asynchronous list is a simple circular list of queue heads. The USB\_ASYNC\_LIST\_ADDR register is simply a pointer to the next queue head. This implements a pure round-robin service for all queue heads linked into the asynchronous list.

52.4.2.3 Isochronous (High-Speed) Transfer Descriptor (iTDR)

The format of an isochronous transfer descriptor is shown in the table below. This structure is used only for high-speed isochronous endpoints. All other transfer types should use queue structures. Isochronous TDs must be aligned on a 32-byte boundary.

Table 52-6. Isochronous Transaction Descriptor (iTDR)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Next Link Pointer																										0		Typ		T	03-00H	
Status				Transaction 0 Length												IO C	PG*		Transaction 0 Offset*												07-04H	
Status				Transaction 1 Length												IO C	PG*		Transaction 1 Offset*												0B-08H	
Status				Transaction 2 Length												IO C	PG*		Transaction 2 Offset*												0F-0CH	

Table continues on the next page...



**Table 52-6. Isochronous Transaction Descriptor (iTID) (continued)**

Status	Transaction 3 Length	IO C	PG*	Transaction 3 Offset*	13-10 H
Status	Transaction 4 Length	IO C	PG*	Transaction 4 Offset*	17-14 H
Status	Transaction 5 Length	IO C	PG*	Transaction 5 Offset*	1B-1 8H
Status	Transaction 6 Length	IO C	PG*	Transaction 6 Offset*	1F-1 CH
Status	Transaction 7 Length	IO C	PG*	Transaction 7 Offset*	23-20 H
Buffer Pointer (Page 0)				EndPt    R    Device Address	27-24 H
Buffer Pointer (Page 1)				I/ O    Maximum Packet Size	2B-2 8H
Buffer Pointer (Page 2)				-    Mult	2F-2 CH
Buffer Pointer (Page 3)				-	33-30 H
Buffer Pointer (Page 4)				-	37-34 H
Buffer Pointer (Page 5)				-	3B-3 8H
Buffer Pointer (Page 6)				-	3F-3 CH



Host Controller Read/Write



Host Controller Read Only

These fields may be modified by the host controller if the I/O field indicates an OUT.

#### 52.4.2.3.1 Next Link Pointer

The first DWord of an iTD is a pointer to the next schedule data structure.

The following table describes the Next Schedule Element pointer field.

**Table 52-7. Next Schedule Element Pointer**

Bit	Description
-----	-------------

*Table continues on the next page...*

**Table 52-7. Next Schedule Element Pointer (continued)**

31-5 Link Pointer (LP)	These bits correspond to memory address signals [31:5], respectively. This field points to another Isochronous Transaction Descriptor (iT/siT) or Queue Head (QH).
4-3 Reserved	These bits are reserved and their value has no effect on operation. Software should initialize this field to zero.
2-1 QH/(s)iTD Select (Typ)	This field indicates to the Host Controller whether the item referenced is an iTD, siTD or a QH. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are:  Value Meaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor) 11b FSTN (frame span traversal node)
0 Terminate (T)	1= Link Pointer field is not valid. 0= Link Pointer field is valid.

### 52.4.2.3.2 iTD Transaction Status and Control List

DWords 1 through 8 are eight slots of transaction control and status.

Each transaction description includes:

- Status results field
- Transaction length (bytes to send for OUT transactions and bytes received for IN transactions).
- Buffer offset. The PG and Transaction X Offset fields are used with the buffer pointer list to construct the starting buffer address for the transaction.

The host controller uses the information in each transaction description plus the endpoint information contained in the first three DWords of the Buffer Page Pointer list, to execute a transaction on the USB.

The following table describes iTD Transaction Status and Control fields.

**Table 52-8. iTD Transaction Status and Control**

Bit	Description
31-28 Status	This field records the status of the transaction executed by the host controller for this slot. This field is a bit vector with the following encoding:
Bit	Definition
31	Active. Set to one by software to enable the execution of an isochronous transaction by the Host Controller. When the transaction associated with this descriptor is completed, the Host Controller sets this bit to zero indicating that a transaction for this element should not be executed when it is next encountered in the schedule.
30	Data Buffer Error. Set to a one by the Host Controller during status update to indicate that the Host Controller is unable to keep up with the reception of incoming data (overflow) or is unable to supply data fast enough during transmission (under run). If an overflow condition occurs, no action is necessary.
29	Babble Detected. Set to one by the Host Controller during status update when "babble" is detected during the transaction generated by this descriptor.
28	Transaction Error (XactErr). Set to one by the Host Controller during status update in the case where the host did not receive a valid response from the device (Timeout, CRC, Bad PID, etc.). This bit may only be set for isochronous IN transactions.
27-16 Transaction X Length	For an OUT, this field is the number of data bytes the host controller sends during the transaction. The host controller is not required to update this field to reflect the actual number of bytes transferred during the transfer. For an IN, the initial value of the endpoint to deliver. During the status update, the host controller writes back the field is the number of bytes the host expects the number of bytes successfully received. The value in this register is the actual byte count (0±zero length data, 1±one byte, 2±two bytes, etc.). The maximum value this field may contain is 0xC00 (3072).
15 Interrupt On Complete (IOC)	If this bit is set to one, it specifies that when this transaction completes, the Host Controller should issue an interrupt at the next interrupt threshold.
14-12 Page Select (PG)	These bits are set by software to indicate which of the buffer page pointers the offset field in this slot should be concatenated to produce the starting memory address for this transaction. The valid range of values for this field is 0 to 6.
11-0 Transaction X Offset	This field is a value that is an offset, expressed in bytes, from the beginning of a buffer. This field is concatenated onto the buffer page pointer indicated in the adjacent PG field to produce the starting buffer address for this transaction.

### 52.4.2.3.3 iTD Buffer Page Pointer List (Plus)

DWords 9-15 of an isochronous transaction descriptor are nominally page pointers (4 K aligned) to the data buffer for this transfer descriptor. This data structure requires the associated data buffer to be contiguous (relative to virtual memory), but allows the physical memory pages to be non-contiguous.

Seven page pointers are provided to support the expression of eight isochronous transfers. The seven pointers allow for 3 (transactions) \* 1024 (maximum packet size) \* 8 (transaction records) (24576 bytes) to be moved with this data structure, regardless of the alignment offset of the first page.

Because each pointer is a 4 K aligned page pointer, the least significant 12 bits in several of the page pointers are used for other purposes.

The tables below illustrate the field descriptions.

**Table 52-9. iTD Buffer Pointer Page 0 (Plus)**

Bit	Description
31-12 Buffer Pointer (Page 0)	This is a 4 K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11-8 Endpoint Number (Endpt)	This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.
7 Reserved	Bit reserved for future use and should be initialized by software to zero.
6-0 Device Address	This field selects the specific device serving as the data source or sink.

**Table 52-10. iTD Buffer Pointer Page 1 (Plus)**

Bit	Description
31-12 Buffer Pointer (Page 1)	This is a 4K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11 Direction (I/O)	0 = OUT; 1 = IN. This field encodes whether the high-speed transaction should use an IN or OUT PID.
10-0 Maximum Packet Size	This directly corresponds to the maximum packet size of the associated endpoint ( <i>wMaxPacketSize</i> ). This field is used for high-bandwidth endpoints where more than one transaction is issued per transaction description (per micro-frame). This field is used with the <i>Multi</i> field to support high-bandwidth pipes. This field is also used for all IN transfers to detect packet babble. Software should not set a value larger than 1024 (400h). Any value larger yields undefined results.

**Table 52-11. iTD Buffer Pointer Page 2 (Plus)**

Bit	Description
31-12 Buffer Pointer	This is a 4K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11-2 Reserved	This bit reserved for future use and should be set to zero.
1-0 Multi	This field is used to indicate to the host controller the number of transactions that should be executed per transaction description (per micro-frame). The valid values are:  Value Meaning 00b Reserved. A zero in this field yields undefined results. 01b One transaction to be issued for this endpoint per micro- frame 10b Two transactions to be issued for this endpoint per micro- frame 11b Three transactions to be issued for this endpoint per micro- frame

**Table 52-12. iTD Buffer Pointer Page 3-6**

Bit	Description
31-12 Buffer Pointer	This is a 4 K aligned pointer to physical memory. Corresponds to memory address bits [31:12].
11-0 Reserved	These bits reserved for future use and should be set to zero.

#### 52.4.2.4 Split Transaction Isochronous Transfer Descriptor (siTD)

All Full-speed isochronous transfers through the internal transaction translator are managed using the siTD data structure. This data structure satisfies the operational requirements for managing the split transaction protocol.

The following table shows the Split Transaction Isochronous Transfer Descriptor (siTD).

**Table 52-13. Split Transaction Isochronous Transfer Descriptor**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Addr
Next Link Pointer																										0		Typ		T	03-00	
I/O	Port Number						-	Hub Addr						Reserved				EndPt				-	Device Address						07-04 <sup>1</sup>			
Reserved												μFrame C-mask						μFrame S-mask						0B-08 <sup>1</sup>								
io c	P	Reserved				Total Bytes to Transfer						μFrame C-prog-mask						Status						0F-0C <sup>2</sup>								
Buffer Pointer (Page 0)																		Current Offset										13-10 <sup>2</sup>				
Buffer Pointer (Page 1)																		Reserved						TP		T-count		17-14 <sup>2</sup>				
Back Pointer																										0		T		1B-18		

1. 04-0B: Static Endpoint State

2. 0C-13: Transfer results



Host Controller Read/Write



Host Controller Read Only

##### 52.4.2.4.1 Next Link Pointer

DWord0 of a siTD is a pointer to the next schedule data structure.

The following table describes the Next Link Pointer fields.

**Table 52-14. Next Link Pointer**

Bit	Description
31-5	Next Link Pointer (LP). This field contains the address of the next data object to be processed in the periodic list and corresponds to memory address signals [31:5], respectively.
4-3	Reserved. These bits must be written as zeros.
2-1	QH/(s)iTD Select (Typ). This field indicates to the Host Controller whether the item referenced is an iTD/siTD or a QH. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are:  Value Meaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor) 11b FSTN (frame span traversal node)
0	Terminate (T). 1=Link Pointer field is not valid. 0=Link Pointer is valid.

#### 52.4.2.4.2 siTD Endpoint Capabilities/Characteristics

DWords 1 and 2 specify static information about the full-speed endpoint, the addressing of the parent Companion Controller, and micro-frame scheduling control.

The tables below describe the Endpoint and transaction translator characteristics and micro-frame schedule control fields.

**Table 52-15. Endpoint and Transaction Translator Characteristics**

Bit	Description
31	Direction (I/O). 0 = OUT; 1 = IN. This field encodes whether the full-speed transaction should be an IN or OUT.
30-24	Port Number. This field is the port number of the recipient Transaction Translator.
23	Reserved. Bit reserved and should be set to zero.
22-16	Hub Address. This field holds the device address of the Companion Controllers' hub.
15-12	Reserved. Field reserved and should be set to zero.
11-8	Endpoint Number (Endpt). This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.
7	Reserved. Bit is reserved for future use. It should be set to zero.
6-0	Device Address. This field selects the specific device serving as the data source or sink.

**Table 52-16. Micro-frame Schedule Control**

Bit	Description
31-16	Reserved. This field reserved for future use. It should be set to zero.

*Table continues on the next page...*

**Table 52-16. Micro-frame Schedule Control (continued)**

Bit	Description
15-8	Split Completion Mask (mFrame C-Mask). This field (along with the <i>Active</i> and <i>SplitX-state</i> fields in the <i>Status</i> byte) is used to determine during which micro-frames the host controller should execute complete-split transactions. When the criteria for using this field is met, an all zeros value has undefined behavior. The host controller uses the value of the three low-order bits of the FRINDEX register to index into this bit field. If the FRINDEX register value indexes to a position where the <i>mFrame C-Mask</i> field is a one, then this siTD is a candidate for transaction execution. There may be more than one bit in this mask set.
7-0	Split Start Mask (mFrame S-mask). This field (along with the <i>Active</i> and <i>SplitX-state</i> fields in the <i>Status</i> byte) is used to determine during which micro-frames the host controller should execute start-split transactions. The host controller uses the value of the three low-order bits of the FRINDEX register to index into this bit field. If the FRINDEX register value indexes to a position where the <i>mFrame S-mask</i> field is a one, then this siTD is a candidate for transaction execution. An all zeros value in this field, in combination with existing periodic frame list has undefined results.

### 52.4.2.4.3 siTD Transfer State

DWords 3-6 are used to manage the state of the transfer.

The following table describes siTD transfer state fields.

**Table 52-17. siTD Transfer Status and Control**

Bit	Description
31	Interrupt On Complete (ioc). 0 = Do not interrupt when transaction is complete. 1 = Do interrupt when transaction is complete. When the host controller determines that the split transaction has completed it asserts a hardware interrupt at the next interrupt threshold.
30	Page Select (P). Used to indicate which data page pointer should be concatenated with the <i>CurrentOffset</i> field to construct a data buffer pointer (0 selects <i>Page 0</i> pointer and 1 selects <i>Page 1</i> ). The host controller is not required to write this field back when the siTD is retired ( <i>Active</i> bit transitioned from a one to a zero).
29-26	Reserved. This field reserved for future use and should be set to zero.
25-16	Total Bytes To Transfer. This field is initialized by software to the total number of bytes expected in this transfer. Maximum value is 1023 (3FFh)
15-8	μFrame Complete-split Progress Mask (C-prog-Mask). This field is used by the host controller to record which split-completes has been executed.
<b>7-0: Status—This field records the status of the transaction executed by the host controller for this slot. It is a bit vector with the encoding shown in the following rows.</b>	
7	Active. Set to one by software to enable the execution of an isochronous split transaction by the Host Controller.
6	ERR. Set to a one by the Host Controller when an ERR response is received from the Companion Controller.
5	Data Buffer Error. Set to a one by the Host Controller during status update to indicate that the Host Controller is unable to keep up with the reception of incoming data (overrun) or is unable to supply data fast enough during transmission (under run). In the case of an under run, the Host Controller transmits an incorrect CRC (thus invalidating the data at the endpoint). If an overrun condition occurs, no action is necessary.
4	Babble Detected. Set to a one by the Host Controller during status update when "babble" is detected during the transaction generated by this descriptor.
3	Transaction Error (XactErr). Set to a one by the Host Controller during status update in the case where the host did not receive a valid response from the device (Timeout, CRC, Bad PID, etc.). This bit is set only for IN transactions.

*Table continues on the next page...*

**Table 52-17. siTD Transfer Status and Control (continued)**

Bit	Description
2	Missed Micro-Frame. The host controller detected that a host-induced hold-off caused the host controller to miss a required complete-split transaction.
1	Split Transaction State (SplitXstate). The bit encodings are:  Value Meaning  00b Do Start Split.  This value directs the host controller to issue a Start split transaction to the endpoint when a match is encountered in the S-mask.  01b Do Complete Split.  This value directs the host controller to issue a Complete split transaction to the endpoint when a match is encountered in the C-mask.
0	Reserved. Bit reserved for future use and should be set to zero.

#### 52.4.2.4.4 siTD Buffer Pointer List (plus)

DWords 4 and 5 are the data buffer page pointers for the transfer. This structure supports one physical page cross. The most significant 20 bits of each DWord in this section are the 4 K (page) aligned buffer pointers.

The least significant 12 bits of each DWord are used as additional transfer state. The following table describes the siTD buffer pointer fields.

**Table 52-18. Buffer Page Pointer List (plus)**

Bit	Description
31-12	Buffer Pointer List. Bits [31:12] of DWords 4 and 5 are 4 K page aligned physical memory addresses. These bits correspond to physical address bits [31:12] respectively. The lower 12 bits in each pointer are defined and used as specified below. The field <i>P</i> (see <a href="#">siTD Transfer State</a> ) specifies the <i>current</i> active pointer.
Bits 11-0 (Page 0)	Current Offset—The 12 least significant bits of the Page 0 pointer are the current byte offset for the current page pointer (as selected with the page indicator bit ( <i>P</i> field)). The host controller is not required to write this field back when the siTD is retired ( <i>Active</i> bit transitioned from a one to a zero).
<b>Bits 11-0 (Page 1)—The least significant bits of the Page 1 pointer are split into three subfields as shown in the following rows.</b>	
11-5 (Page 1)	Reserved

*Table continues on the next page...*



**Table 52-18. Buffer Page Pointer List (plus) (continued)**

Bit	Description
4-3 (Page 1)	Transaction position (TP). This field is used with T-count to determine whether to send <i>all</i> , <i>first</i> , <i>middle</i> , or <i>last</i> with each outbound transaction payload. System software must initialize this field with the appropriate starting value. The host controller must correctly manage this state during the lifetime of the transfer. The bit encodings are:  Value Meaning  00b All. The entire full-speed transaction data payload is in this transaction (that is, less than or equal to 188 bytes). 01b Begin. This is the first data payload for a full-speed that is greater than 188 bytes. 10b Mid. This is the <i>middle</i> payload for a full-speed OUT transaction that is larger than 188 bytes. 11b End. This is the <i>last</i> payload for a full-speed OUT transaction that was larger than 188 bytes.
2-0 (Page 1)	Transaction count (T-Count). Software initializes this field with the number of OUT start-splits this transfer requires. Any value larger than 6 is undefined.

#### 52.4.2.4.5 siTD Back Link Pointer

DWord 6 of a siTD is simply another schedule link pointer. This pointer is always zero, or references a siTD, and it cannot reference any other schedule data structure.

The following table describes the siTD back link pointer fields

.

**Table 52-19. siTD Back Link Pointer**

Bit	Description
31-5	siTD Back Pointer. This field is a physical memory pointer to a siTD.
4-1	Reserved. This field is reserved for future use. It should be set to zero.
0	Terminate (T). 1 = siTD Back Pointer field is not valid. 0 = siTD Back Pointer field is valid.

#### 52.4.2.5 Queue Element Transfer Descriptor (qTD)

This data structure is only used with a queue head. It describes one or more USB transactions to transfer up to 20480 (5\*4096) bytes.

The structure contains two structure pointers used for queue advancement, a DWord of transfer state, and a five-element array of data buffer pointers.

It is 32 bytes and must be physically contiguous.

The buffer associated with this transfer must be virtually contiguous. The buffer may start on any byte boundary; however, for optimal utilization of on-chip busses it is recommended to align the buffers on a 32-byte boundary. A separate buffer pointer list element must be used for each physical page in the buffer, regardless of whether the buffer is physically contiguous.

Host controller updates (host controller writes) to stand-alone qTDs only occur during transfer retirement. References in the following bit field definitions of updates to the qTD are to the qTD portion of a queue head.

The following table shows the queue head data structure.

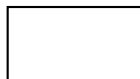
**Table 52-20. Queue Head Data Structure**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Addr																										
Next qTD Pointer																												0																												T	03-00	
Alternate Next qTD Pointer																												0																													T	07-04
dt	Total Bytes to Transfer															io c	C_Page	Cerr	PID Code	Status										0B-08 <sup>1</sup>																												
Buffer Pointer (page 0)																				Current Offset										0F-0C <sup>1</sup>																												
Buffer Pointer (page 0)																				Reserved										13-10																												
Buffer Pointer (page 0)																				Reserved										17-14																												
Buffer Pointer (page 0)																				Reserved										1B-18																												
Buffer Pointer (page 0)																				Reserved										1F-1C																												

1. 08-0F: Transfer Results



Host Controller Read/Write



Host Controller Read Only

Queue Element Transfer Descriptors must be aligned on 32-byte boundaries.

### 52.4.2.5.1 Next qTD Pointer

The first DWord of an element transfer descriptor is a pointer to another transfer element descriptor.

The following table describes Next qTD pointer fields.

**Table 52-21. qTD Next Element Transfer Pointer (DWord 0)**

Bit	Description
31-5	Next Transfer Element Pointer. This field contains the physical memory address of the next qTD to be processed. The field corresponds to memory address signals[31:5], respectively.
4-1	Reserved
0	Terminate (T). 1= pointer is invalid. 0=Pointer is valid (points to a valid Transfer Element Descriptor). This bit indicates to the Host Controller that there are no more valid entries in the queue.

### 52.4.2.5.2 Alternate Next qTD Pointer

The second DWord of a queue element transfer descriptor is used to support hardware-only advance of the data stream to the next transfer descriptor on short packet. To be more explicit the host controller always uses this pointer when the current qTD is retired due to short packet.

The following table describes the TD Alternate Next Element Transfer Pointer field descriptions.

**Table 52-22. TD Alternate Next Element Transfer Pointer (DWord 1)**

Bit	Description
31-5	Alternate Next Transfer Element Pointer. This field contains the physical memory address of the next qTD to be processed in the event that the current qTD execution encounters a short packet (for an IN transaction). The field corresponds to memory address signals [31:5], respectively.
4-1	Reserved
0	Terminate (T). 1= pointer is invalid. 0=Pointer is valid (points to a valid Transfer Element Descriptor). This bit indicates to the Host Controller that there are no more valid entries in the queue.

### 52.4.2.5.3 qTD Token

The third DWord of a queue element transfer descriptor contains most of the information the host controller requires to execute a USB transaction (the remaining endpoint-addressing information is specified in the queue head).

**NOTE**

The field descriptions forward reference fields defined in the queue head. Where necessary, these forward references are preceded with a QH notation.

The following table describes the TD Token fields.

**Table 52-23. TD Token (DWord 2)**

Bit	Description
31 Data Toggle	This is the data toggle sequence bit. The use of this bit depends on the setting of the <i>Data Toggle Control</i> bit in the queue head.
30-16 Total Bytes to Transfer	This field specifies the total number of bytes to be moved with this transfer descriptor. This field is decremented by the number of bytes actually moved during the transaction, only on the successful completion of the transaction. The maximum value software may store in this field is 5 * 4K (5000H). This is the maximum number of bytes 5 page pointers can access. If the value of this field is zero when the host controller fetches this transfer descriptor (and the active bit is set), the host controller executes a zero-length transaction and retires the transfer descriptor. It is not a requirement for OUT transfers that <i>Total Bytes To Transfer</i> be an even multiple of QHD.Maximum Packet Length. If software builds such a transfer descriptor for an OUT transfer, the last transaction is always less than QHD.Maximum Packet Length.  Although it is possible to create a transfer up to 20K this assumes the 1 <sup>st</sup> offset into the first page is 0. When the offset cannot be predetermined, crossing past the 5th page can be guaranteed by limiting the total bytes to 16K**. Therefore, the maximum recommended transfer is 16 K(4000H).
15 Interrupt On Complete (IOC)	If this bit is set to a one, it specifies that when this qTD is completed, the Host Controller should issue an interrupt at the next interrupt threshold.
14-12 Current Page (C_Page)	This field is used as an index into the qTD buffer pointer list. Valid values are in the range 0H to 4H. The host controller is not required to write this field back when the qTD is retired.

*Table continues on the next page...*

**Table 52-23. TD Token (DWord 2) (continued)**

Bit	Description	
11-10 Error Counter (CERR)	<p>This field is a 2-bit down counter that keeps track of the number of consecutive Errors detected while executing this qTD. If this field is programmed with a non-zero value during set-up, the Host Controller decrements the count and writes it back to the qTD if the transaction fails. If the counter counts from one to zero, the Host Controller marks the qTD inactive, sets the <i>Halted</i> bit to a one, and error status bit for the error that caused <i>CERR</i> to decrement to zero. An interrupt is generated if the <i>USB Error Interrupt Enable</i> bit in the <i>USBINTR</i> register is set to a one. If HCD programs this field to zero during set-up, the Host Controller does not count errors for this qTD and there is no limit on the retries of this qTD. Note that write-backs of intermediate execution state are to the queue head overlay area, not the qTD.</p> <p>Transaction Error - Decrement  Data Buffer Error - No Decrement<sup>3</sup>  Stalled - No Decrement<sup>1</sup>  Babble Detected - No Decrement<sup>1</sup>  No Error - No Decrement<sup>2</sup></p>	
	Error	Decrement Counter
	1	Detection of Babble or Stall automatically halts the queue head. Thus, count is not decremented
	2	<p>If the <i>EPS</i> field indicates a HS device or the queue head is in the Asynchronous Schedule (and <i>PIDCode</i> indicates an IN or OUT) and a bus transaction completes and the host controller does not detect a transaction error, then the host controller should reset <i>CERR</i> to extend the total number of errors for this transaction. For example, <i>CERR</i> should be reset with maximum value (3) on each successful completion of a transaction. The host controller must never reset this field if the value at the start of the transaction is 00b.</p> <p>See <a href="#">Split Transaction Interrupt</a> for CERR adjustment rules when the <i>EPS</i> field indicates a FS or LS device and the queue head is in the Periodic Schedule. See <a href="#">Asynchronous - Do Complete Split</a> for CERR adjustment rules when the <i>EPS</i> field indicates a FS or LS device, the queue head is in the Asynchronous schedule and the <i>PIDCode</i> indicates a SETUP.</p>
	3	Data buffer errors are host problems. They don't count against the device's retries.
	<p><b>NOTE:</b> Software must not program CERR to a value of zero when the <i>EPS</i> field is programmed with a value indicating a Full- or Low-speed device. This combination could result in undefined behavior.</p>	
9-8 PID Code	<p>This field is an encoding of the token, which should be used for transactions associated with this transfer descriptor. Encodings are:</p>	
	00b	OUT Token generates token (E1H)
	01b	IN Token generates token (69H)
	10b	SETUP Token generates token (2DH) (undefined if endpoint is an interrupt, the queue head is non-zero) transfer type, for example, <i>μFrame S-mask</i> field in
	11b	Reserved

Table continues on the next page...

**Table 52-23. TD Token (DWord 2) (continued)**

Bit	Description
7-0 Status	This field is used by the Host Controller to communicate individual command execution states back to HCD. This field contains the status of the last transaction performed on this qTD. The bit encodings are:
Bit	Status Field Description
7	Active. Set to one by software to enable the execution of transactions by the Host Controller.
6	Halted. Set to one by the Host Controller during status updates to indicate that a serious error has occurred at the device/endpoint addressed by this qTD. This can be caused by babble, the error counter counting down to zero, or reception of the STALL handshake from the device during a transaction. Any time that a transaction results in the Halted bit being set to a one, the Active bit is also set to zero.
5	Data Buffer Error. Set to a one by the Host Controller during status update to indicate that the Host Controller is unable to keep up with the reception of incoming data (overflow) or is unable to supply data fast enough during transmission (under run). If an overflow condition occurs, the Host Controller forces a timeout condition on the USB, invalidating the transaction at the source. If the host controller sets this bit to a one, then it remains a one for the duration of the transfer.
4	Babble Detected. Set to a one by the Host Controller during status update when "babble" is detected during the transaction. In addition to setting this bit, the Host Controller also sets the <i>Halted</i> bit to a one. Because "babble" is considered a fatal error for the transfer, setting the Halted bit to a one insures that no more transactions occur because of this descriptor.
3	Transaction Error (XactErr). Set to a one by the Host Controller during status update in the case where the host did not receive a valid response from the device (Timeout, CRC, Bad PID, etc.). If the host controller sets this bit to a one, then it remains a one for the duration of the transfer.
2	Missed Micro-Frame. This bit is ignored unless the <i>QH.EPS</i> field indicates a full- or low-speed endpoint and the queue head is in the periodic list. This bit is set when the host controller detected that a host-induced hold-off caused the host controller to miss a required complete-split transaction. If the host controller sets this bit to a one, then it remains a one for the duration of the transfer.
1	Split Transaction State (SplitXstate). This bit is ignored by the host controller unless the <i>QH.EPS</i> field indicates a full- or low-speed endpoint. When a Full- or Low-speed device, the host controller uses this bit to track the state of the split-transaction. The functional requirements of the host controller for managing this state bit and the split transaction protocol depends on whether the endpoint is in the periodic or asynchronous schedule. The bit encodings are:  Value Meaning 0b Do Start Split. This value directs the host controller to issue a Start split transaction to the endpoint. 1b Do Complete Split. This value directs the host controller to issue a Complete split transaction to the endpoint.
0	Ping State (P)/ERR. If the <i>QH.EPS</i> field indicates a High-speed device and the <i>PID_Code</i> indicates an OUT endpoint, then this is the state bit for the Ping protocol. The bit encodings are:  Value Meaning 0b Do OUT. This value directs the host controller to issue an OUT PID to the endpoint. 1b Do Ping. This value directs the host controller to issue a PING PID to the endpoint.  If the <i>QH.EPS</i> field does not indicate a High-speed device, then this field is used as an error indicator bit. It is set to a one by the host controller whenever a periodic split-transaction receives an ERR handshake.

#### 52.4.2.5.4 qTD Buffer Page Pointer List

The last five DWords of a queue element transfer descriptor is an array of physical memory address pointers. These pointers reference the individual pages of a data buffer.

System software initializes Current Offset field to the starting offset into the current page, where current page is selected through the value in the C\_Page field.

The following table describes the qTD Buffer Pointer(s) (DWords 3-7) fields.

**Table 52-24. qTD Buffer Pointer(s) (DWords 3-7)**

Bit	Description
31-12	Buffer Pointer List. Each element in the list is a 4 K page aligned physical memory address. The lower 12 bits in each pointer are reserved (except for the first one), as each memory pointer must reference the start of a 4 K page. The field C_Page specifies the current active pointer. When the transfer element descriptor is fetched, the starting buffer address is selected using C_Page (similar to an array index to select an array element). If a transaction spans a 4K buffer boundary, the host controller must detect the page-span boundary in the data stream, increment C_Page and advance to the next buffer pointer in the list, and conclude the transaction through the new buffer pointer.
11-0	Current Offset (Reserved). This field is reserved in all pointers except the first one (for example Page 0). The host controller should ignore all reserved bits. For the page 0 current offset interpretation, this field is the byte offset into the current page (as selected by C_Page). The host controller is not required to write this field back when the qTD is retired. Software should ensure the Reserved fields are initialized to zero.

#### 52.4.2.6 Queue Head

The following table shows the queue head structure layout.

**Table 52-25. Queue Head Structure Layout**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Addr
Queue Head Horizontal Link Pointer																										0		Typ		T	03-00	
RL				C	Maximum Packet Length										H	dt c	EP	EndPt				I	Device Address						07-04 <sup>1</sup>			
Mult		Port Number <sup>2</sup>							Hub Addr <sup>2</sup>							µFrame C-mask <sup>2</sup>							µFrame S-mask							0B-08 <sup>1</sup>		
Current qTD Pointer																										0				0F-0C		
Next qTD Pointer																										0				T	13-10 <sup>3</sup>	
Alternate Next qTD pointer																												NakC nt		T	17-14 <sup>4</sup>	

*Table continues on the next page...*

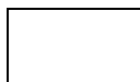
**Table 52-25. Queue Head Structure Layout (continued)**

dt	Total Bytes to Transfer	io c	C_Page	Cerr	PID Code	Status	1B- 18
	Buffer Pointer (Page 0)				Current Offset		1F- 1C
	Buffer Pointer (Page 1)				Reserved	C-prog-mask <sup>2</sup>	23- 20
	Buffer Pointer (Page 2)				S-bytes <sup>2</sup>	FrameTa g <sup>2</sup>	27- 24 <sup>4</sup>
	Buffer Pointer (Page 3)				Reserved		2B- 28
	Buffer Pointer (Page 4)				Reserved		2F- 2C <sup>3</sup>

1. 04-0B: Static endpoint state.
2. These fields are used exclusively to support split transactions to USB 2.0 hubs
3. 10-2F: Transfer overlay.
4. 14-27: Transfer results.



Host Controller Read/Write



Host Controller Read Only

#### 52.4.2.6.1 Queue Head Horizontal Link Pointer

The first DWord of a Queue Head contains a link pointer to the next data object to be processed after any required processing in this queue has been completed, as well as the control bits defined below.

This pointer may reference a queue head or one of the isochronous transfer descriptors. It must not reference a queue element transfer descriptor.

The following table describes the Queue head DWord 0 fields.

**Table 52-26. Queue Head DWord 0**

Bit	Description
31-5	Queue Head Horizontal Link Pointer (QHLP). This field contains the address of the next data object to be processed in the horizontal list and corresponds to memory address signals [31:5], respectively.
4-3	Reserved
2-1	QH/(s)iTD Select (Typ). This field indicates to the hardware whether the item referenced by the link pointer is an iTD, siTD or a QH. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are:  Value Meaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor) 11b FSTN (frame span traversal node)

*Table continues on the next page...*



**Table 52-26. Queue Head DWord 0 (continued)**

0	Terminate (T). 1=Last QH (pointer is invalid). 0=Pointer is valid. If the queue head is in the context of the periodic list, a one bit in this field indicates to the host controller that this is the end of the periodic list. This bit is ignored by the host controller when the queue head is in the Asynchronous schedule. Software must ensure that queue heads reachable by the host controller always have valid horizontal link pointers.
---	---

#### 52.4.2.6.2 Queue Head Endpoint Capabilities/Characteristics

The second and third DWords of a Queue Head specifies static information about the endpoint. This information does not change over the lifetime of the endpoint.

There are three types of information in this region:

- Endpoint Characteristics. These are the USB endpoint characteristics including addressing, maximum packet size, and endpoint speed.
- Endpoint Capabilities. These are adjustable parameters of the endpoint. They effect how the endpoint data stream is managed by the host controller.
- Split Transaction Characteristics. This data structure is used to manage full- and low-speed data streams for bulk, control, and interrupt via split transactions to USB2.0 Hub Transaction Translator. There are additional fields used for addressing the hub and scheduling the protocol transactions (for periodic).

The host controller must not modify the bits in this region.

The following table describes the Endpoint characteristics: Queue head DWord 1 fields.

**Table 52-27. Endpoint Characteristics: Queue Head DWord 1**

Bit	Description
31-28	Nak Count Reload (RL). This field contains a value, which is used by the host controller to reload the Nak Counter field.
27	Control Endpoint Flag (C). If the <i>QH.EPS</i> field indicates the endpoint is not a high-speed device, and the endpoint is a control endpoint, then software must set this bit to a one. Otherwise, it should always set this bit to zero.
26-16	Maximum Packet Length. This directly corresponds to the maximum packet size of the associated endpoint ( <i>wMaxPacketSize</i> ). The maximum value this field may contain is 0x400 (1024).
15	Head of Reclamation List Flag (H). This bit is set by System Software to mark a queue head as being the head of the reclamation list.
14	Data Toggle Control (DTC). This bit specifies where the host controller should get the initial data toggle on an overlay transition.  0b Ignore DT bit from incoming qTD. Host controller preserves DT bit in the queue head.  1b Initial data toggle comes from incoming qTD DT bit. Host controller replaces DT bit in the queue head from the DT bit in the qTD.

*Table continues on the next page...*

**Table 52-27. Endpoint Characteristics: Queue Head DWord 1 (continued)**

13-12	Endpoint Speed (EPS). This is the speed of the associated endpoint. Bit combinations are:	
	Value	Meaning
	00b	Full-Speed (12 Mbits/sec)
	01b	Low-Speed (1.5 Mbits/sec)
	10b	High-Speed (480 Mbits/sec)
	11b	Reserved
This field must not be modified by the host controller.		
11-8	Endpoint Number (Endpt). This 4-bit field selects the particular endpoint number on the device serving as the data source or sink.	
7	Inactivate on Next Transaction (I). This bit is used by system software to request that the host controller set the Active bit to zero. See <a href="#">Rebalancing the periodic schedule</a> , for full operational details. This field is only valid when the queue head is in the Periodic Schedule and the <i>EPS</i> field indicates a Full or Low-speed endpoint. Setting this bit to one when the queue head is in the Asynchronous Schedule or the <i>EPS</i> field indicates a high-speed device yields undefined results.	
6-0	Device Address. This field selects the specific device serving as the data source or sink.	

The table below describes the Endpoint capabilities: Queue head DWord 2 field descriptions.

**Table 52-28. Endpoint Capabilities: Queue Head DWord 2**

Bit	Description
31-30	High-Bandwidth Pipe Multiplier (Mult). This field is a multiplier used to key the host controller as the number of successive packets the host controller may submit to the endpoint in the current execution. The host controller makes the simplifying assumption that software properly initializes this field (regardless of location of queue head in the schedules or other run time parameters). The valid values are:  Value Meaning 00b Reserved. A zero in this field yields undefined results. 01b One transaction to be issued for this endpoint per micro-frame 10b Two transactions to be issued for this endpoint per micro-frame 11b Three transactions to be issued for this endpoint per micro-frame
29-23	Port Number. This field is ignored by the host controller unless the <i>EPS</i> field indicates a full- or low-speed device. The value is the port number identifier on the USB 2.0 Hub (for hub at device address <i>Hub Addr</i> below), below which the full- or low-speed device associated with this endpoint is attached. This information is used in the split-transaction protocol.
22-16	Hub Addr. This field is ignored by the host controller unless the <i>EPS</i> field indicates a full- or low-speed device. The value is the USB device address of the USB 2.0 Hub below which the full- or low-speed device associated with this endpoint is attached. This field is used in the split-transaction protocol.
15-8	Split Completion Mask ( $\mu$ Frame C-Mask). This field is ignored by the host controller unless the <i>EPS</i> field indicates this device is a low- or full-speed device and this queue head is in the periodic list. This field (along with the <i>Active</i> and <i>SplitX-state</i> fields) is used to determine during which micro-frames the host controller should execute a complete-split transaction. When the criteria for using this field are met, a zero value in this field has undefined behavior. This field is used by the host controller to match against the three low-order bits of the FRINDEX register. If the FRINDEX register bits decode to a position where the $\mu$ Frame C- Mask field is a one, then this queue head is a candidate for transaction execution. There may be more than one bit in this mask set.

Table continues on the next page...

**Table 52-28. Endpoint Capabilities: Queue Head DWord 2 (continued)**

7-0	Interrupt Schedule Mask ( $\mu$ Frame S-mask). This field is used for all endpoint speeds. Software should set this field to a zero when the queue head is on the asynchronous schedule. A non-zero value in this field indicates an interrupt endpoint. The host controller uses the value of the three low-order bits of the FRINDEX register as an index into a bit position in this bit vector. If the $\mu$ Frame S-mask field has a one at the indexed bit position then this queue head is a candidate for transaction execution. If the EPS field indicates the endpoint is a high-speed endpoint, then the transaction executed is determined by the PID_Code field contained in the execution area. This field is also used to support split transaction types: Interrupt (IN/OUT). This condition is true when this field is non-zero and the EPS field indicates this is either a full- or low-speed device. A zero value in this field, in combination with existing in the periodic frame list has undefined results.
-----	---

### 52.4.2.6.3 Transfer Overlay-Queue Head

The nine DWords in this area represent a transaction working space for the host controller. The general operational model is that the host controller can detect whether the overlay area contains a description of an active transfer. If it does not contain an active transfer, then it follows the Queue Head Horizontal Link Pointer to the next queue head. The host controller will never follow the Next Transfer Queue Element or Alternate Queue Element pointers unless it is actively attempting to advance the queue. For the duration of the transfer, the host controller keeps the incremental status of the transfer in the overlay area. When the transfer is complete, the results are written back to the original queue element.

The DWord3 of a Queue Head contains a pointer to the source qTD currently associated with the overlay. The host controller uses this pointer to write back the overlay area into the source qTD after the transfer is complete.

The following table describes the current qTD link pointer field descriptions.

**Table 52-29. Current qTD Link Pointer**

Bit	Description
31-5	Current Element Transaction Descriptor Link Pointer. This field contains the address Of the current transaction being processed in this queue and corresponds to memory address signals [31:5], respectively.
4-0	Reserved (R). These bits are ignored by the host controller when using the value as an address to write data. The actual value may vary depending on the usage.

The DWords 4-11 of a queue head are the transaction overlay area. This area has the same base structure as a Queue Element Transfer Descriptor. The queue head utilizes the reserved fields of the page pointers to implement tracking the state of split transactions.

This area is characterized as an overlay because when the queue is advanced to the next queue element, the source queue element is merged onto this area. This area serves as execution cache for the transfer.

The table below describes the Host-controller rules for bits in overlay.

**Table 52-30. Host-Controller Rules for Bits in Overlay (DWords 5, 6, 8 and 9)**

DWord	Bit	Description
5	4-1	Nak Counter (NakCnt) $\mu$ RW. This field is a counter the host controller decrements whenever a transaction for the endpoint associated with this queue head results in a Nak or Nyet response. This counter is reloaded from <i>RL</i> before a transaction is executed during the first pass of the reclamation list (relative to an Asynchronous List Restart condition). It is also loaded from <i>RL</i> during an overlay.
6	31	Data Toggle. The <i>Data Toggle Control</i> controls whether the host controller preserves this bit when an overlay operation is performed.
6	15	Interrupt On Complete (IOC). The IOC control bit is always inherited from the source <i>qTD</i> when the overlay operation is performed.
6	11-10	Error Counter (C_ERR). This two-bit field is copied from the <i>qTD</i> during the overlay and written back during queue advancement.
6	0	Ping State (P)/ERR. If the <i>EPS</i> field indicates a high-speed endpoint, then this field should be preserved during the overlay operation.
8	7-0	Split-transaction Complete-split Progress (C-prog-mask). This field is initialized to zero during any overlay. This field is used to track the progress of an interrupt split-transaction.
9	4-0	Split-transaction Frame Tag (Frame Tag). This field is initialized to zero during any overlay. This field is used to track the progress of an interrupt split-transaction.
9	11-5	S-bytes. Software must ensure that the <i>S-bytes</i> field in a <i>qTD</i> is zero before activating the <i>qTD</i> . This field is used to keep track of the number of bytes sent or received during an IN or OUT split transaction.

### 52.4.2.7 Periodic Frame Span Traversal Node (FSTN)

This data structure is to be used only for managing Full- and Low-speed transactions that span a Host-frame boundary.

See [Host Controller Operational Model for FSTNs](#) for full operational details. Software must not use an FSTN in the Asynchronous Schedule. An FSTN in the Asynchronous schedule results in undefined behavior. Software must not use the FSTN feature with a host controller whose USB\_HCIVERSION register indicates a revision implementation below 0096h. FSTNs are not defined for implementations before 0.96 and their use yields undefined results.

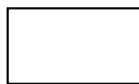
**Table 52-31. Frame Span Traversal Node Structure Layout**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Addr
Normal Path Link Pointer																										0		Typ		T	03-00	
Back Path Link Pointer																										0		Typ <sup>1</sup>		T	07-04	

1. Must be set to indicate a queue head



Host Controller Read/Write



Host Controller Read Only

### 52.4.2.7.1 FSTN Normal Path Pointer

The first DWord of an FSTN contains a link pointer to the next schedule object. This object can be of any valid periodic schedule data type.

The following table describes the FSTN normal path pointer fields.

**Table 52-32. FSTN Normal Path Pointer Field Descriptions**

Bit	Description
31-5	Normal Path Link Pointer (NPLP). This field contains the address of the next data object to be processed in the periodic list and corresponds to memory address signals [31:5], respectively.
4-3	Reserved
2-1	QH/(s)iTD/FSTN Select (Typ). This field indicates to the Host Controller whether the item referenced is a iTD/ siTD, a QH or an FSTN. This allows the Host Controller to perform the proper type of processing on the item after it is fetched. Value encodings are:  Value Meaning 00b iTD (isochronous transfer descriptor) 01b QH (queue head) 10b siTD (split transaction isochronous transfer descriptor) 11b FSTN (Frame Span Traversal Node)
0	Terminate (T). 1=Link Pointer field is not valid. 0=Link Pointer is valid.

### 52.4.2.7.2 FSTN Back Path Link Pointer

The second DWord of an FTSN node contains a link pointer to a queue head.

If the T-bit in this pointer is zero, then this FSTN is a Save-Place indicator. Its Typ field must be set by software to indicate the target data structure is a queue head. If the T-bit in this pointer is set to one, then this FSTN is the Restore indicator. When the T-bit is one, the host controller ignores the Typ field.

The following table describes the FSTN back path link pointer fields.

**Table 52-33. FSTN Back Path Link Pointer Field Descriptions**

Bit	Description
31-5	Back Path Link Pointer (BPLP). This field contains the address of a Queue Head. This field corresponds to memory address signals [31:5], respectively.
4-3	Reserved
2-1	Typ. Software must ensure this field is set to indicate the target data structure is a Queue Head. Any other value in this field yields undefined results.

*Table continues on the next page...*

**Table 52-33. FSTN Back Path Link Pointer Field Descriptions  
(continued)**

0	<p>Terminate (T). 1=Link Pointer field is not valid (that is the host controller must not use bits [31:5] as a valid memory address). This value also indicates that this FSTN is a Restore indicator.</p> <p>0=Link Pointer is valid (that is the host controller may use bits [31:5] (in combination with the CTRLDSSEGMENT register if applicable) as a valid memory address). This value also indicates that this FSTN is a Save-Place indicator.</p>
---	---

### 52.4.3 Host Operational Model

The general operational model is for the enhanced interface host controller hardware and enhanced interface host controller driver (generally referred to as system software).

Each significant operational feature of the EHCI host controller is discussed in a separate section. Each section presents the operational model requirements for the host controller hardware. Where appropriate, recommended system software operational models for features are also presented.

#### 52.4.3.1 Host Controller Initialization

After initial power-on or HCRreset (hardware or through HCRreset bit in the USB\_USBCMD register), all of the operational registers are at their default values. After a hardware reset, only the operational registers not contained in the Auxiliary power well are at their default values.

The following table describes the default values of operational registers.

**Table 52-34. Default Values of Operational Register Space**

Operational Register	Default Value (after Reset)
USB_USBCMD	00080000h (00080B00h, if <i>Asynchronous Schedule Park Capability</i> is one)
USB_USBSTS	00001000h
USB_USBINTR	00000000h
USB_FRINDEX	00000000h
USB_CTRLDSSEGMENT	00000000h
USB_PERIODICLISTBASE	Undefined
USB_ASYNC_LISTADDR	Undefined
USB_CONFIGFLAG	00000000h
USB_PORTSC1	00002000h (w/PPC set to one); 00003000h (w/PPC set to zero)

To initialize the host controller, software should perform the following steps:

- Write the appropriate value to the USB\_USBINTR register to enable the appropriate interrupts.
- Write the base address of the Periodic Frame List to the USB\_PERIODICLIST BASE register. If no work items are in the periodic schedule, all elements of the Periodic Frame List should have their T-Bits set to one.
- Write the USB\_USBCMD register to set the desired interrupt threshold, frame list size (if applicable) and turn the host controller ON through setting the Run/Stop bit.

At this point, the host controller is up and running and the port registers begin reporting device connects, and so on. System software can enumerate a port through the reset process (where the port is in the enabled state). At this point, the port is active with SOFs occurring down the enabled ports, but the schedules have not enabled. To communicate with devices through the asynchronous schedule, system software must write the USB\_ASYNCLISTADDR register with the address of a control or bulk queue head. Software must then enable the asynchronous schedule by writing one to the Asynchronous Schedule Enable bit in the USB\_USBCMD register. To communicate with devices through the periodic schedule, system software must enable the periodic schedule by writing one to the Periodic Schedule Enable bit in the USB\_USBCMD register.

#### NOTE

The schedules can be turned on before the first port is reset (and enabled).

When the USB\_USBCMD register is written, system software must ensure the appropriate bits are preserved, depending on the intended operation.

### 52.4.3.2 Port Routing and Control

The EHCI specification defines that a USB 2.0 Host controller is comprised of one high-speed host controller, which implements the EHCI programming interface and 0 to N USB 1.1 companion host controllers.

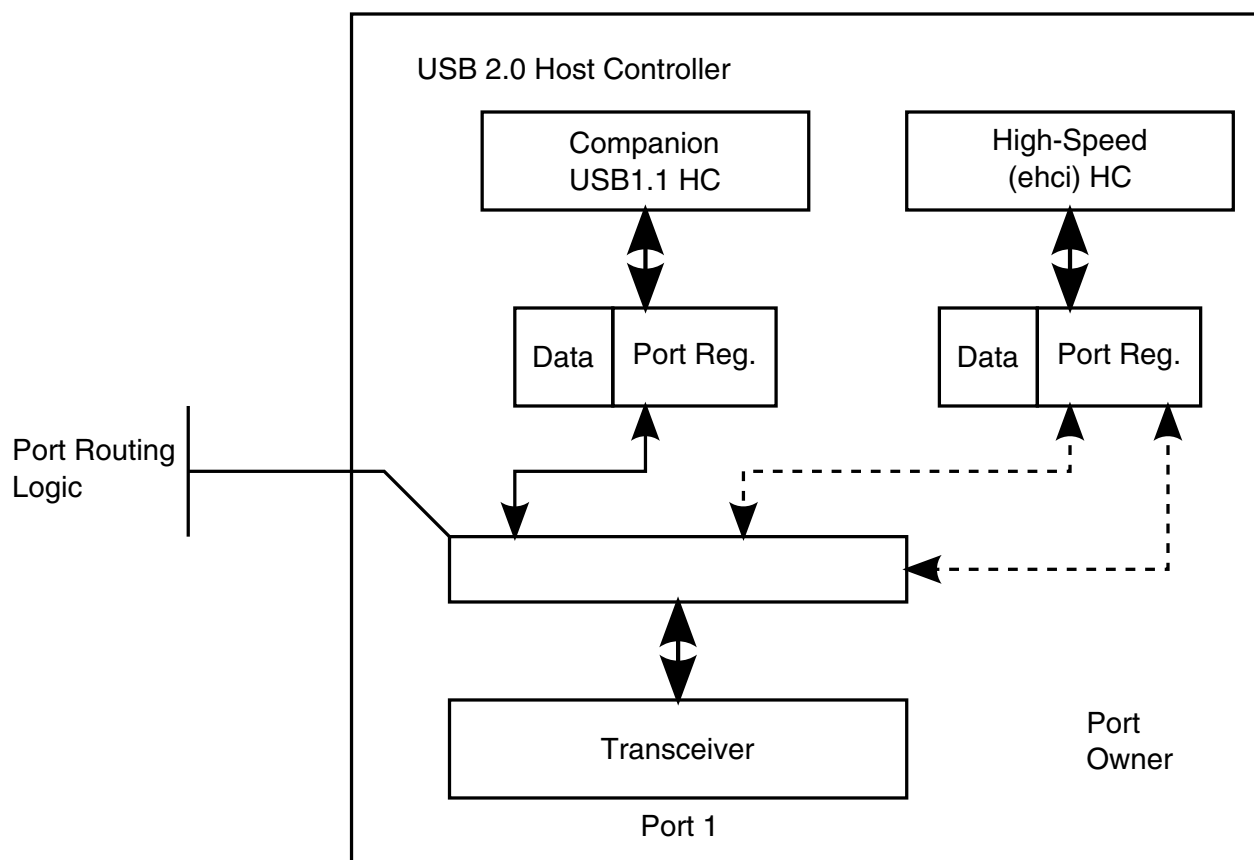
Companion host controllers (cHCs) may be implementations of either Universal or Open host controller specifications. This configuration is used to deliver the required full USB 2.0-defined port capability; for example, Low-, Full-, and High-speed capability for every port.

#### NOTE

The USB controllers on i.MX parts do not require nor support companion controllers to support Full and Low Speed device. Full and Low Speed devices are supported within the USB

controller by emulating the functionality of a high-speed HUB. Therefore, no port routing is present in the controller. Please refer to [Embedded Transaction Translator Function](#) for detail!

The following figure illustrates a simple block diagram of the port routing logic and its relationship to the high-speed and companion host controllers within a USB 2.0 host controller.



**Figure 52-5. Example USB 2.0 Host Controller Port Routing Block Diagram**

There exists one transceiver per physical port and each host controller block has its own port status and control registers. The EHCI controller has port status and control registers for every port. Each companion host controller has only the port control and status registers it is required to operate. Either the EHCI host controller or one companion host controller controls each transceiver. Routing logic lies between the transceiver, the port status and control registers.<sup>1</sup>

The port routing logic is controlled from signals originating in the EHCI host controller. The EHCI host controller has a global routing policy control field and per-port ownership control fields. The Configured Flag (CF) bit is the global routing policy control. At

1. The routing logic should not be implemented in the 480 MHz clock domain of the transceiver.



power-on or reset, the default routing policy is to the companion controllers (if they exist). If the system does not include a driver for the EHCI host controller and the host controller includes Companion Controllers, then the ports still work in Full- and Low-speed mode (assuming the system includes a driver for the companion controllers). In general, when the EHCI owns the ports, the companion host controllers' port registers do not see a connect indication from the transceiver. Similarly, when a companion host controller owns a port, the EHCI controller's port registers do not see a connect indication from the transceiver. The details on the rules for the port routing logic are described in the following sections. The USB 2.0 host controller must be implemented as a multi-function PCI device if the implementation includes companion controllers. The companion host controllers' function numbers must be less than the EHCI host controller function number. The EHCI host controller must be a larger function number with respect to the companion host controllers associated with this EHCI host controller. If a PCI device implementation contains only an EHCI controller (that is no companion controllers or other PCI functions), then the EHCI host controller must be function zero, in accordance with the PCI Specification. The N\_CC field in the Structural Parameter register (HCSPARAMS) indicates whether the controller implementation includes companion host controllers. When N\_CC has a non-zero value there exists companion host controllers. If N\_CC has a value of zero, then the host controller implementation does not include companion host controllers. If the host controller root ports are exposed to attachment of full- or low-speed devices, the ports always fails the high-speed chirp during reset and the ports are not enabled. System software can notify the user of the illegal condition. This type of implementation requires a USB 2.0 hub be connected to a root port to provide full and low-speed device connectivity.

System software uses information in the host controller capability registers to determine how the ports are routed to the companion host controllers. See [Host Controller Structural Parameters \(USBC\\_n\\_HCSPARAMS\)](#).

#### 52.4.3.2.1 Port Routing Control through EHCI Configured (CF) Bit

Each port in the USB 2.0 host controller are routed either to a single companion host controller or to the EHCI host controller.

The port routing logic is controlled by two mechanisms in the EHCI HC: a host controller global flag and per-port control. The Configured Flag (CF) bit, is used to globally set the policy of the routing logic. Each port register has a Port Owner control bit which allows the EHCI Driver to explicitly control the routing of individual ports. Whenever the CF bit transitions from zero to one (this transition is only available under program control) the port routing unconditionally routes all of the port registers to the EHCI HC (all Port Owner bits go to zero). While the CF-bit is one, the EHCI Driver controls individual ports' routing through the Port Owner control bit. Likewise, whenever the CF bit

transitions from one to zero (as a result of Aux power application, HCRESET, or software writing zero to CF-bit), the port routing unconditionally routes all of the port registers to the appropriate companion HC. The default value for the EHCI HC's CF bit (after Aux power application or HCRESET) is zero.

The *view* of the port depends on the current owner. A Universal or Open companion host controller will see port register bits consistent with the appropriate specification. Port bit definitions that are required for EHCI host controllers are not visible to companion host controllers.

The following table summarizes the default routing for all the ports, based on the value of the EHCI HC's CF bit.

**Table 52-35. Default Port Routing Depending on EHCI HC CF Bit**

HS CF Bit	Default Port Ownership	Explanation
0B	Companion HCs	The companion host controllers own the ports and only Full- and Low-speed devices are supported in the system. The exact port assignments are implementation dependent. The ports behave only as Full- and Low-speed ports in this configuration
1B	EHCI HC	The EHCI host controller has default ownership over all of the ports. The routing logic inhibits device connect events from reaching the companion HCs' port status and control registers when the port owner is the EHCI HC. The EHCI HC has access to the additional port status and control bits defined in this specification (see <a href="#">Port Status &amp; Control (USBC <i>n</i> PORTSC1)</a> ). The EHCI HC can temporarily release control of the port to a companion HC by setting the <i>PortOwner</i> bit in the PORTSC1 register to one.

### 52.4.3.2.2 Port Routing Control through PortOwner and Disconnect Event

Manipulating the port routing through the CF-bit is an extreme process and not intended to be used during normal operation.

The normal mode of port ownership transferal is on the granularity of individual ports using the Port Owner bit in the EHCI HC's USB\_PORTSC1 register (for hand-offs from EHCI to companion host controllers). Individual port ownership is returned to the EHCI controller when the port registers a device disconnect. When the disconnect is detected, the port routing logic immediately returns the port ownership to the EHCI controller. The companion host controller port register detects the device disconnect and operates normally.

Under normal operating conditions (assuming all HC drivers loaded and operational and the EHCI *CF-bit* is set to one), the typical port enumeration sequence proceeds as illustrated below:

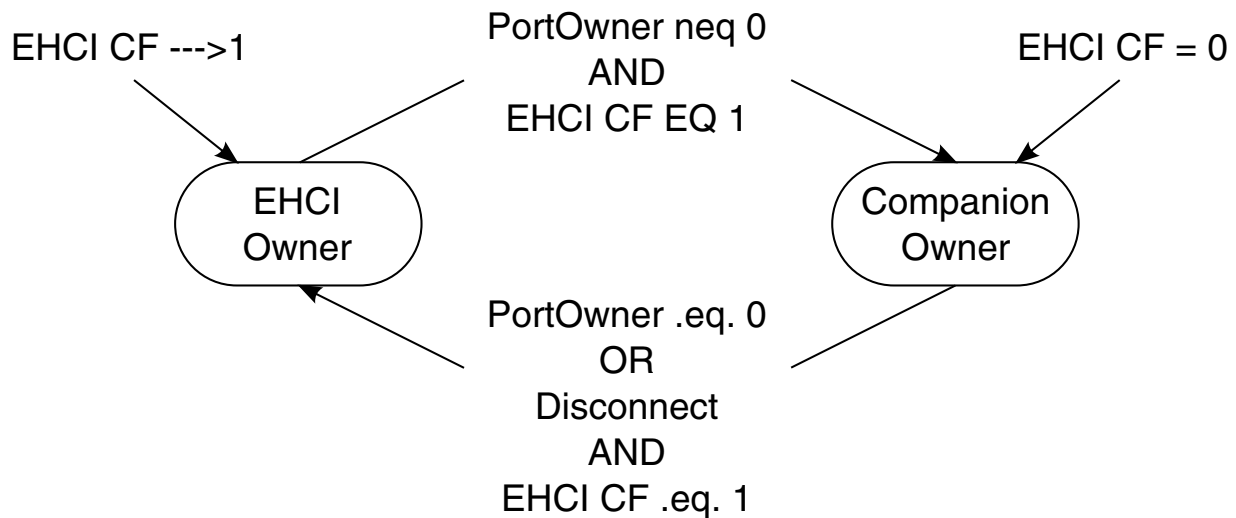
- Initial condition is that EHCI is port owner. A device is connected causing the port to detect a connect, set the port connect change bit and issue a port-change interrupt (if enabled).
- EHCI Driver identifies the port with the new connect change bit asserted and sends a change report to the hub driver. Hub driver issues a GetPortStatus() request and identifies the connect change. It then issues a request to clear the connect change, followed by a request to reset and enable the port.
- When the EHCI Driver receives the request to reset and enable the port, it first checks the value reported by the LineStatus bits in the USB\_PORTSC1 register. If they indicate the attached device is a full-speed device (for example, D+ is asserted), then the EHCI Driver sets the PortReset control bit to one (and sets the PortEnable bit to zero) which begins the reset-process. Software times the duration of the reset, then terminates reset signaling by writing zero to the port reset bit. The reset process is actually complete when software reads zero in the PortReset bit. The EHCI Driver checks the PortOwner bit in the USB\_PORTSC1 register. If set to one, the connected device is a high-speed device and EHCI Driver (root hub emulator) issues a change report to the hub driver and the hub driver continues to enumerate the attached device.
- At the time the EHCI Driver receives the port reset and enable request the LineStatus bits might indicate a low-speed device. Additionally, when the port reset process is complete, the PortEnable field may indicate that a full-speed device is attached. In either case the EHCI driver sets the PortOwner bit in the USB\_PORTSC1 register to one to release port ownership to a companion host controller.
- When the EHCI Driver sets PortOwner bit to one, the port routing logic makes the connection state of the transceiver available to the companion host controller port register and removes the connection state from the EHCI HC port. The EHCI USB\_PORTSC1 register observes and reports a disconnect event through the disconnect change bit. The EHCI Driver detects the connection status change (either by polling or by port change interrupt) and then sends a change report to the hub driver. When the hub driver requests that port-state, the EHCI Driver responds with a reset complete change set to one, a connect change set to one and a connect status set to zero. This information is derived directly from the EHCI port register. This allows the hub driver to assume the device was disconnected during reset. It acknowledges the change bits and wait for the next change event. While the EHCI controller does not own the port, it simply remains in a state where the port reports no device connected. The device-connect evaluation circuitry of the companion HC activates and detects the device, the companion Driver detects the connection and enumerates the port.

When a port is routed to a companion HC, it remains under the control of the companion HC until the device is disconnected from the root port (ignoring for now the scenario where EHCI's CF-bit transitions from 1b to 0b). When a disconnect occurs, the disconnect

event is detected by both the companion HC port control and the EHCI port ownership control. On the event, the port ownership is returned immediately to the EHCI controller. The companion HC stack detects the disconnect and acknowledges as it would in an ordinary standalone implementation. Subsequent connects is detected by the EHCI port register and the process repeats.

### 52.4.3.2.3 Example Port Routing State Machine

The following figure illustrates an example of how the port ownership should be managed. The following sections describe the entry conditions to each state.



**Figure 52-6. Port Owner Handoff State Machine**

#### 52.4.3.2.3.1 EHCI HC Owner

Entry to this state occurs when one of the following events occur:

- When the EHCI HC's Configure Flag (CF) bit in the USB\_CONFIGFLAG register transitions from zero to one. This signals the fact that the system has a host controller driver for the EHCI HC and that all ports in the USB 2.0 host controller must default route to the EHCI controller.
- When the port is owned by a companion HC and the device is disconnected from the port. The EHCI port routing control logic is notified of the disconnect, and returns port routing to the EHCI controller. The connection state of the companion HC goes immediately to the disconnected state (with appropriate side effect to connect change, enable and enable change). The companion HC driver acknowledges the disconnect by setting the connect status change bit to zero. This allows the

companion HC's driver to interact with the port completely through the disconnect process.

- When system software writes zero to the PortOwner bit in the USB\_PORTSC1 register. This allows software to take ownership of a port from a companion host controller. When this occurs, the routing logic to the companion HC effectively signals a disconnect to the companion HC's port status and control register.

#### 52.4.3.2.3.2 Companion HC Owner

Entry to this state occurs whenever one of the following events occur:

- When the PortOwner field transitions from zero to one.
- When the HS-mode HC's Configure Flag (CF) is equal to zero.

On entry to this state, the routing logic allows the companion HC port register to detect a device connect. Normal port enumeration proceeds.

#### 52.4.3.2.4 Port Power

The Port Power Control (PPC) bit in the USB\_HCSPARAMS register indicates whether the USB 2.0 host controller has port power control (see [Host Controller Structural Parameters \(USBC\\_n\\_HCSPARAMS\)](#)).

When this bit is zero, then the host controller does not support software control of port power switches. When in this configuration, the port power is always available and the companion host controllers must implement functionality consistent with port power always on. When the *PPC* bit is one, then the host controller implementation includes port power switches. Each available switch has an output enable, which is referred to in this discussion as PortPowerOutputEnable (PPE). PPE is controlled based on the state of the combination bits PPC bit, EHCI Configured (CF)-bit and individual Port Power (PP) bits.

The following table describes the summary behavioral model.

**Table 52-36. Port Power Enable Control Rules**

CF	CHC <sup>1</sup> (PP)	EHCI <sup>2</sup> (PP)	Owner	PPE <sup>3</sup>	Description
0	0	X	CHC	0	When the EHCI controller is not configured, the port is owned by the companion host controller. When the companion HC's port power select is off, then the port power is off.
0	1	X	CHC	1	Similar to previous entry. When the companion HC's port power select is on, then the port power is on.

*Table continues on the next page...*

**Table 52-36. Port Power Enable Control Rules (continued)**

1	0	0	CHC	0	Port owner has port power turned off, the power to port is off.
1	0	0	EHC	0	Port owner has port power turned off, the power to port is off.
1	0	1	EHC	1	Port owner has port power on, so power to port is on.
1	0	1	CHC	1	If either HC has port power turned on, the power to the port is on.
1	1	0	EHC	1	If either HC has port power turned on, the power to the port is on.
1	1	0	CHC	1	Port owner has port power on, so power to port is on.
1	1	1	CHC	1	Port owner has port power on, so power to port is on.
1	1	1	EHC	1	Port owner has port power on, so power to port is on.

1. CHC (Companion Host Controller).
2. EHC (EHCI Host Controller).
3. PPE (Port Power Enable). This bit actually turns on the port power switch (if one exists).

#### 52.4.3.2.5 Port Reporting Over-Current

Host controllers are by definition power providers on USB. Whether the ports are considered high- or low-powered is a platform implementation issue. Each EHCI USB\_PORTSC1 register has an over-current status and over-current change bit.

The functionality of these bits are specified in the USB Specification Revision 2.0.

The over current detection and limiting logic usually resides outside the host controller logic. This logic may be associated with one or more ports. When this logic detects an over-current condition it is made available to both the companion and EHCI ports. The effect of an over-current status on a companion host controller port is beyond the scope of this document.

The over-current condition effects the following bits in the USB\_PORTSC1 register on the EHCI port:

- Over-current Active bits are set to one. When the over-current condition goes away, the Over-current Active bit transitions from one to zero.
- Over-current Change bits are set to one. On every transition of the Over-current Active bit the host controller sets the Over-current Change bit to one. Software sets the Over-current Change bit to zero by writing one to this bit.

- Port Enabled/Disabled bit is set to zero. When this change bit gets set to one, then the Port Change Detect bit in the USB\_USBSTS register is set to one.
- Port Power (PP) bits may optionally be set to zero. There is no requirement in USB that a power provider shut off power in an over current condition. It is sufficient to limit the current and leave power applied. When the Over-current Change bit transitions from zero to one, the host controller also sets the Port Change Detect bit in the USB\_USBSTS register to one. In addition, if the Port Change Interrupt Enable bit in the USB\_USBINTR register is one, then the host controller issues an interrupt to the system. Refer to [Table 52-37](#) for summary behavior for over-current detection when the host controller is halted (suspended from a device component point of view).

### 52.4.3.3 Suspend/Resume-Host Operational Model

The EHCI host controller provides an equivalent suspend and resume model as that defined for individual ports in a USB 2.0 Hub.

Control mechanisms are provided to allow system software to suspend and resume individual ports. The mechanisms allow the individual ports to be resumed completely through software initiation. Other control mechanisms are provided to parameterize the host controller's response (or sensitivity) to external resume events. In this discussion, host-initiated, or software initiated resumes are called Resume Events/Actions. Bus-initiated resume events are called wake-up events. The classes of wake-up events are:

- Remote-wake-up enabled device asserts resume signaling. In similar kind to USB 2.0 Hubs, EHCI controllers must always respond to explicit device resume signaling and wake-up the system (if necessary).
- Port connect and disconnect and over-current events. Sensitivity to these events can be turned on or off by using the per-port control bits in the USB\_PORTSC1 registers.

Selective suspend is a feature supported by every USB\_PORTSC1 register. It is used to place specific ports into a suspend mode. This feature is used as a functional component for implementing the appropriate power management policy implemented in a particular operating system. When system software intends to suspend the entire bus, it should selectively suspend all enabled ports, then shut off the host controller by setting the Run/Stop bit in the USB\_USBCMD register to zero. The EHCI sub-block can then be placed into a lower device state through the PCI power management interface (see Appendix A, Enhanced Host Controller Interface Specification for Universal Serial Bus, Revision 0.95, November 2000, Intel Corporation. <http://www.intel.com>).

When a wake event occurs, the system resumes operation and system software eventually set the Run/Stop bit to one and resume the suspended ports. Software must not set the Run/Stop bit to one until it is confirmed that the clock to the host controller is stable. This is usually confirmed in a system implementation in that all of the clocks in the system are stable before the ARM platform is restarted. So, by definition, if software is running, clocks in the system are stable and the Run/Stop bit in the USB\_USBCMD register can be set to one. Minimum system software delays are also defined in the PCI Power Management Specification. Refer to PCI Power Management Specification for more information.

#### 52.4.3.3.1 Port Suspend/Resume

System software places individual ports into suspend mode by writing one into the appropriate USB\_PORTSC1 Suspend bit. Software must only set the Suspend bit when the port is in the enabled state (Port Enabled bit is one) and the EHCI is the port owner (PortOwner bit is zero).

The host controller may evaluate the Suspend bit immediately or wait until a micro-frame or frame boundary occurs. If evaluated immediately, the port is not suspended until the current transaction (if one is executing) completes. Therefore, there may be several micro-frames of activity on the port until the host controller evaluates the Suspend bit. The host controller must evaluate the Suspend bit at least every frame boundary.

System software can initiate a resume on a selectively suspended port by writing one to the Force Port Resume bit. Software should not attempt to resume a port unless the port reports that it is in the suspended state (see [Port Status & Control \(USBC<sub>n</sub>\\_PORTSC1\)](#)). If system software sets Force Port Resume bit to one when the port is not in the suspended state, the resulting behavior is undefined. In order to assure proper USB device operation, software must wait for at least 10 ms after a port indicates that it is suspended (Suspend bit is one) before initiating a port resume through the Force Port Resume bit. When Force Port Resume bit is one, the host controller sends resume signaling down the port. System software times the duration of the resume (nominally 20 ms) then sets the Force Port Resume bit to zero. When the host controller receives the write to transition Force Port Resume to zero, it completes the resume sequence as defined in the USB specification, and sets both the Force Port Resume and Suspend bits to zero. Software-initiated port resumes do not affect the Port Change Detect bit in the USB\_USBSTS register nor do they cause an interrupt if the Port Change Interrupt Enable bit in the USB\_USBINTR register is one. An external USB event may also initiate a resume. The wake events are defined above. When a wake event occurs on a suspended port, the resume signaling is detected by the port and the resume is reflected downstream



within 100  $\mu$ sec. The port's Force Port Resume bit is set to one and the Port Change Detect bit in the USB\_USBSTS register is set to one. If the Port Change Interrupt Enable bit in the USB\_USBINTR register is one the host controller issues a hardware interrupt.

System software observes the resume event on the port, delays a port resume time (nominally 20 ms), then terminates the resume sequence by writing zero to the Force Port Resume bit in the port. The host controller receives the write of zero to Force Port Resume, terminates the resume sequence and sets Force Port Resume and Suspend port bits to zero. Software can determine that the port is enabled (not suspended) by sampling the USB\_PORTSC1 register and observing that the Suspend and Force Port Resume bits are zero. Software must ensure that the host controller is running (that is HCHalted bit in the USB\_USBSTS register is zero), before terminating a resume by writing zero to a port's Force Port Resume bit. If HCHalted is one when Force Port Resume is set to zero, then SOFs do not occur down the enabled port and the device returns to suspend mode in a maximum of 10 msec.

The table below summarizes the wake-up events. Whenever a resume event is detected, the Port Change Detect bit in the USB\_USBSTS register is set to one. If the Port Change Interrupt Enable bit is one in the USB\_USBINTR register, the host controller generates an interrupt on the resume event. Software acknowledges the resume event interrupt by clearing the Port Change Detect status bit in the USB\_USBSTS register.

**Table 52-37. Behavior During Wake-up Events**

Port Status and Signaling Type	Signaled Port Response	Device State	
		D0	Not D0
Port disabled, resume K-State received	No Effect	N/A	N/A
Port suspended, resume K-State received	Resume reflected downstream on signaled port. Force Port Resume status bit in USB_PORTSC1 register is set to one. Port Change Detect bit in USB_USBSTS register set to one.	[1], [2]	[2]
Port is enabled, disabled or suspended, and the port's WKDSCNNT_E bit is one. A disconnect is detected.	Depending in the initial port state, the USB_PORTSC1 Connected Enable status bits are set to zero, and the Connect Change status bit is set to one. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [2]	[2]
Port is enabled, disabled or suspended, and the port's WKDSCNNT_E bit is zero. A disconnect is detected.	Depending on the initial port state, the USB_PORTSC1 Connect and Enable status bits are set to zero, and the Connect Change status bit is set to one. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [3]	[3]
Port is not connected and the port's WKCNTNT_E bit is one. A connect is detected.	USB_PORTSC1 Connect Status and Connect Status Change bits are set to one. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [2]	[2]
Port is not connected and the port's WKCNTNT_E bit is zero. A connect is detected.	USB_PORTSC1 Connect Status and Connect Status Change bits are set to one. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [3]	[3]

*Table continues on the next page...*

**Table 52-37. Behavior During Wake-up Events (continued)**

Port is connected and the port's WKOC_E bit is one. An over-current condition occurs.	USB_PORTSC1 Over-current Active, Over-current Change bits are set to one. If Port Enable/Disable bit is one, it is set to zero. Port Change Detect bit in the USB_USBSTS register is set to one	[1], [2]	[2]
Port is connected and the port's WKOC_E bit is zero. An over-current condition occurs.	USB_PORTSC1 Over-current Active, Over-current Change bits are set to one. If Port Enable/Disable bit is one, it is set to zero. Port Change Detect bit in the USB_USBSTS register is set to one.	[1], [3]	[3]

[1] Hardware interrupt issued if Port Change Interrupt Enable bit in the USB\_USBINTR register is one.

[2] PME# asserted if enabled (Note: PME Status must always be set to one).

[3] PME# not asserted.

#### 52.4.3.4 Schedule Traversal Rules

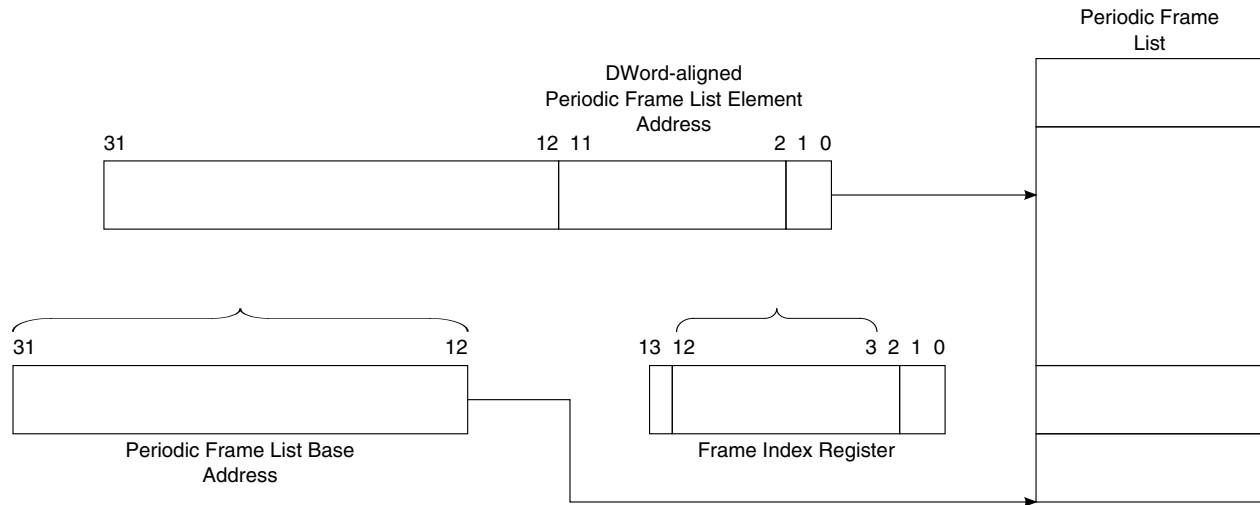
The host controller executes transactions for devices using a simple, shared-memory schedule.

The schedule is comprised of a few data structures, organized into two distinct lists. The data structures are designed to provide the maximum flexibility required by USB, minimize memory traffic and hardware / software complexity.

System software maintains two schedules for the host controller: a periodic schedule and an asynchronous schedule. The root of the periodic schedule is the USB\_PERIODICLISTBASE register (see [Frame List Base Address \(USBC\\_n\\_PERIODICLISTBASE\)/Device Address \(USBC\\_n\\_DEVICEADDR\)](#)). The USB\_PERIODICLISTBASE register is the physical memory base address of the periodic frame list. The periodic frame list is an array of physical memory pointers. The objects referenced from the frame list must be valid schedule data structures as defined in [Host Data Structures](#). In each micro-frame, if the periodic schedule is enabled (see [Periodic scheduling threshold](#)) then the host controller must execute from the periodic schedule before executing from the asynchronous schedule. It only executes from the asynchronous schedule after it encounters the end of the periodic schedule. The host controller traverses the periodic schedule by constructing an array offset reference from the USB\_PERIODICLISTBASE and the USB\_FRINDEX registers (see the following figure). It fetches the element and begins traversing the graph of linked schedule data structures.

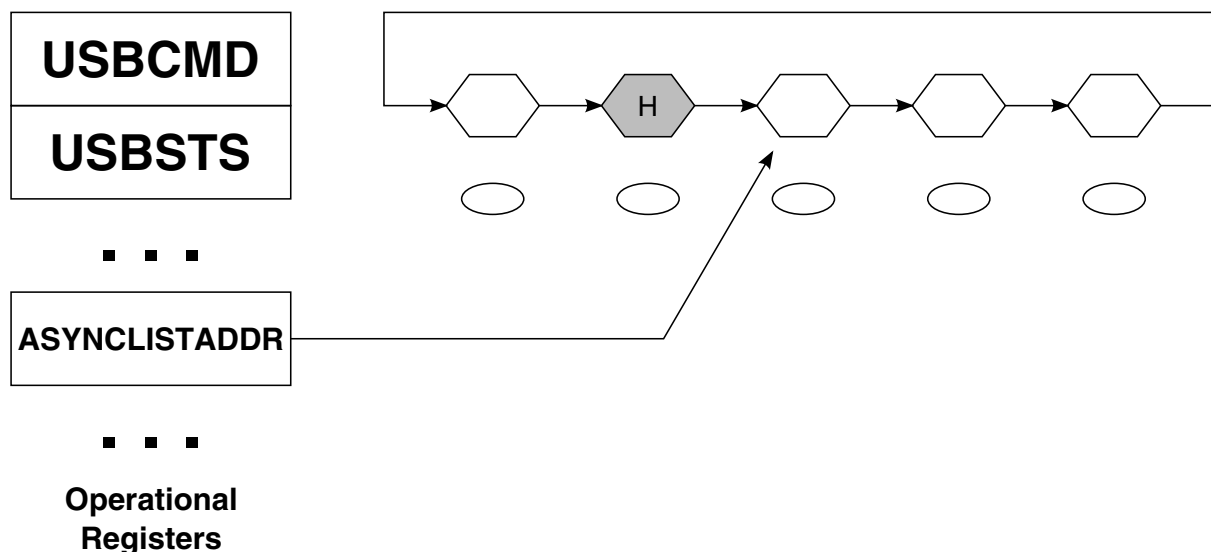
The end of the periodic schedule is identified by a next link pointer of a schedule data structure having its T-bit set to one. When the host controller encounters a T-Bit set to one during a horizontal traversal of the periodic list, it interprets this as an End-Of-Periodic-List mark. This causes the host controller to cease working on the periodic schedule and transitions immediately to traversing the asynchronous schedule. After the transition, the host controller executes from the asynchronous schedule until the end of the micro-frame.

The following figure illustrates the derivation of pointer into frame list array.



**Figure 52-7. Derivation of Pointer into Frame List Array**

When the host controller determines that it is the time to execute from the asynchronous list, it uses the operational register `USB_ASYNC_LIST_ADDR` to access the asynchronous schedule, see the figure below.



**Figure 52-8. General Format of Asynchronous Schedule List**

The USB\_ASYNC\_LIST\_ADDR register contains a physical memory pointer to the next queue head. When the host controller makes a transition to executing the asynchronous schedule, it begins by reading the queue head referenced by the USB\_ASYNC\_LIST\_ADDR register. Software must set queue head horizontal pointer T-bits to zero for queue heads in the asynchronous schedule. See [Asynchronous Schedule](#) for complete operational details.

#### 52.4.3.4.1 Example - Preserving Micro-Frame Integrity

One of the requirements of a USB host controller is to maintain Frame Integrity. This means that the HC must preserve the micro-frame boundaries.

For example, SOF packets must be generated on time (within the specified allowable jitter), and High-speed EOF1,2 thresholds must be enforced. The end of micro-frame timing points EOF1 and EOF2 are clearly defined in the USB Specification Revision 2.0. One implication of this responsibility is that the HC must ensure that it does not start transactions that do not complete before the end of the micro-frame. More precisely, no transactions should be started by the host controller, which do not complete in their entirety before the EOF1 point. In order to enforce this rule, the host controller must check each transaction before it starts to ensure that it completes before the end of the micro-frame.

So, what exactly needs to be involved in this check? Fundamentally, the transaction data payload, plus bit stuffing, plus transaction overhead must be taken into consideration. It is possible to be extremely accurate on how much time the next transaction takes. Take OUTs for an example. The host controller must fetch all of the OUT data from memory

in order to send it onto the USB bus. A host controller implementation could pre-fetch all of the OUT data, and pre-compute the actual number of bits in the token and data packets. In addition, the system knows the depth of the target endpoint, so it could closely estimate turnaround time for handshake. In addition, the host controller knows the size of a handshake packet. Pre-computing effects of bit stuffing and summing up the other overhead numbers can allow the host controller to know exactly whether there is enough bus time, before EOF1 to complete the OUT transaction. To accomplish this particular approach takes an inordinate amount of time and hardware complexity.

The alternative is to make a reasonable guess whether the next transaction can be started. An example approximation algorithm is described below. This example algorithm relies on the EHCI policy that periodic transactions are scheduled first in the micro-frame. It is a reasonable assumption that software never over-commits the micro-frame to periodic transactions greater than the specification allowable 80%. In the available remaining 20% bandwidth, the host controller has some ability (in this example) to decide whether or not to execute a transaction. The result of this algorithm is that sometimes, under some circumstances a transaction is not executed that could have been executed. However, under all circumstances, a transaction is never started unless there is enough time in the frame to complete the transaction.

#### 52.4.3.4.1.1 Transaction Fit - A Best-Fit Approximation Algorithm

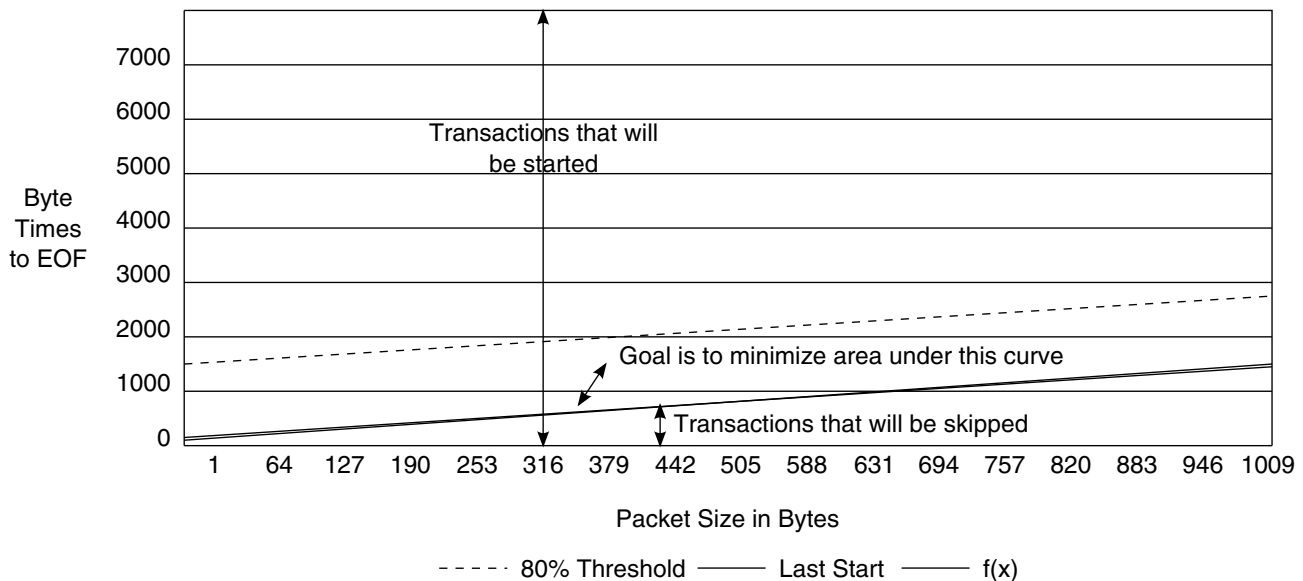
A curve is calculated which represents the latest start time for every packet size, at which software schedules the start of a periodic transaction.

This curve is the 80% bandwidth curve. Another curve is calculated which is the absolute, latest permitted start time for every packet size. This curve represents the absolute latest time, that a transaction of each packet size can be started and completed, in the micro-frame. A plot of these two curves are illustrated in [Figure 52-9](#). The plot Y-axis represents the number of byte-times left in a frame.

The space between the 80% and the Last Start plots is bandwidth reclamation area. In this algorithm the host controller may skip transactions during this time if it is prudent.

The Best-Fit Approximation method plots a function ( $f(x)$ ) between the 80% and Last Start curves. The function  $f(x)$  adds a constant to every transaction's maximum packet size and the result compared with the number of bytes left in the frame. The constant represents an approximation of the effects of bit stuffing and protocol overhead. The host controller starts transactions whose results land above the function curve. The host controller will not start transactions whose results land below the function curve.

The following figure illustrates the Best-Fit Approximation.



**Figure 52-9. Best Fit Approximation**

The LastStart line was calculated in this example to assume the absolute worst-case bus overhead per transaction. The particular transaction used is a start-split, zero-length OUT transaction with a handshake. Summaries of the component parts are listed in the table below. The component times were derived from the protocol timings defined in the USB Specification Revision 2.0.

**Table 52-38. Example Worse-case Transaction Timing Components**

Component	Bit time	Byte Time	Explanation
Split Token	76	9.5	Split token as defined in USB core specification. Includes sync, token, eop, and so on.
Host 2 Host IPG	88	11	Number of bit times required between consecutive host packets.
Token	67	8.375	Token as defined in USB core specification. Includes sync, token, eop, and so on.
Host 2 Host IPG	88	11	Token as defined in USB core specification. Includes sync, token, eop, and so on.
Data Packet (0 data bytes)	66.7	8.34	Zero-length data packet. Includes sync, PID, crc16, eop, and so on.
Turnaround time	721	90.125	Time for packet initiator (Host) to see the beginning of a response to a transmitted packet.
Handshake packet	48	6	Handshake packet as defined in USB core specification. Includes sync, PID, eop, and so on.
		144	Total

The exact details of the function ( $f(x)$ ) are up to the particular implementation. However, it should be obvious that the goal is to minimize the area under the curve between the approximation function and the Last Start curve, without dipping below the LastStart

line, while at the same time keeping the check as simple as possible for hardware implementation. The  $f(x)$  in Figure 52-9 was constructed using the following pseudo-code test on each transaction size data point. This algorithm assumes that the host controller keeps track of the remaining bits in the frame.

```

Algorithm CheckTransactionWillFit (MaximumPacketSize, HC_BytesLeftInFrame)
Begin
  Local Temp = MaximumPacketSize + 192
  Local rvalue = TRUE
  If MaximumPacketSize >= 128 then
    Temp += 128
  End If
  If Temp > HC_BytesLeftInFrame then
    Rvalue = FALSE
  End If
  Return rvalue
End

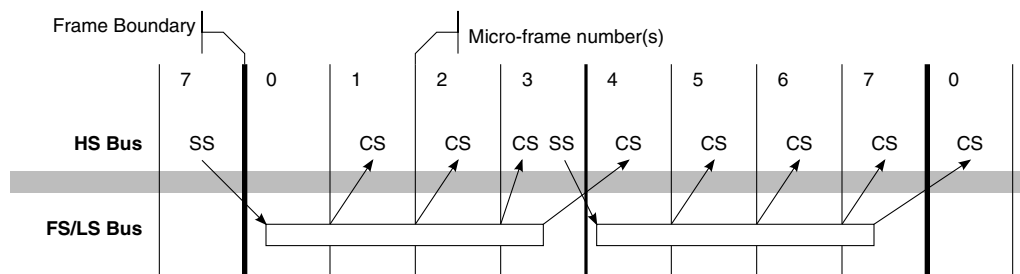
```

This algorithm takes two inputs, the current maximum packet size of the transaction and the hardware counter of the number of bytes left in the current micro-frame. It unconditionally adds a simple constant of 192 to the maximum packet size to account for a first-order effect of transaction overhead and bit stuffing. If the transaction size is greater than or equal to 128 bytes, then an additional constant of 128 is added to the running sum to account for the additional worst-case bit stuffing of payloads larger than 128. An inflection point was inserted at 128 because the  $f(x)$  plot was getting close to the LastStart line.

### 52.4.3.5 Periodic Schedule Frame Boundaries vs Bus Frame Boundaries

The USB Specification Revision 2.0 requires that the frame boundaries (SOF frame number changes) of the high-speed bus and the full- and low-speed bus(s) below USB 2.0 Hubs be strictly aligned.

Super-imposed on this requirement is that USB 2.0 Hubs manage full- and low-speed transactions through a micro-frame pipeline (see start- (SS) and complete- (CS) splits illustrated in the following figure). A simple, direct projection of the frame boundary model into the host controller interface schedule architecture creates tension (complexity for both hardware and software) between the frame boundaries and the scheduling mechanisms required to service the full- and low-speed transaction translator periodic pipelines.



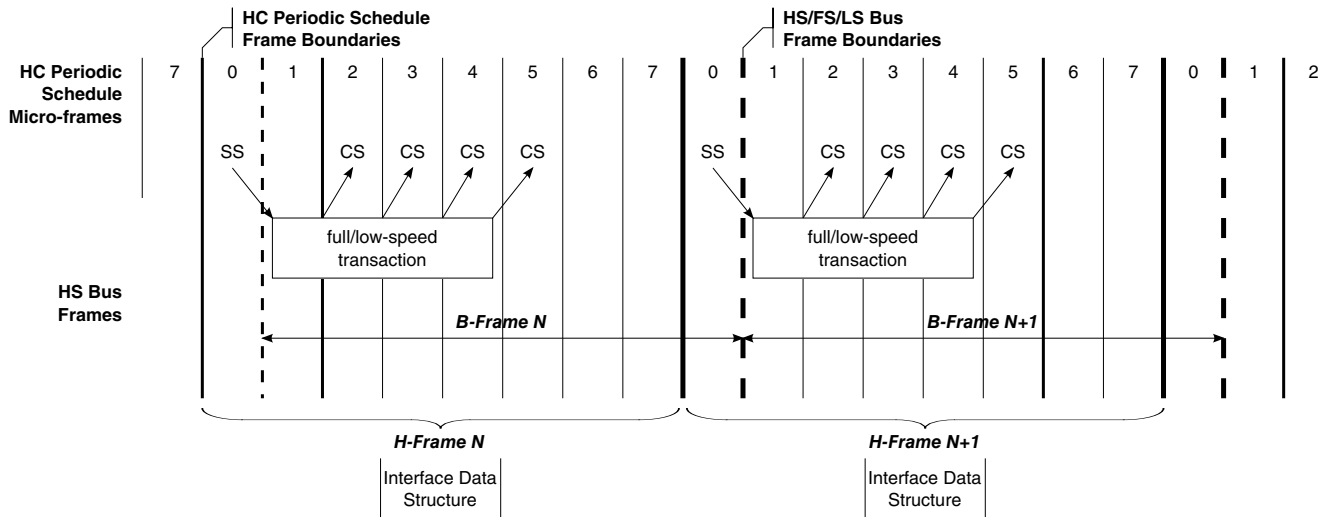
**Figure 52-10. Frame Boundary Relationship between HS bus and FS/LS Bus**

The simple projection, as the above figure illustrates, introduces frame-boundary wrap conditions for scheduling on both the beginning and end of a frame. In order to reduce the complexity for hardware and software, the host controller is required to implement one micro-frame phase shift for its view of frame boundaries. The phase shift eliminates the beginning of frame and frame-wrap scheduling boundary conditions.

The implementation of this phase shift requires that the host controller use one register value for accessing the periodic frame list and another value for the frame number value included in the SOF token. These two values are separate, but tightly coupled. The periodic frame list is accessed through the Frame List Index Register (USB\_FRINDEX) documented in [USB Frame Index \(USBC\\_n\\_FRINDEX\)](#) and initially illustrated in [Schedule Traversal Rules](#). Bits FRINDEX[2:0], represent the micro-frame number. The SOF value is coupled to the value of FRINDEX[13:3]. Both FRINDEX[13:3] and the SOF value are increment based on FRINDEX[2:0]. It is required that the SOF value be delayed from the FRINDEX value by one micro-frame. The one micro-frame delay yields host controller periodic schedule and bus frame boundary relationship as illustrated in the following figure. This adjustment allows software to trivially schedule the periodic start and complete-split transactions for full-and low-speed periodic endpoints, using the natural alignment of the periodic schedule interface. The reasons for selecting this phase-shift are beyond the scope of this specification.

The following figure illustrates how periodic schedule data structures relate to schedule frame boundaries and bus frame boundaries. To aid the presentation, two terms are defined: The host controller's view of the 1 msec boundaries is called H-Frames. The high-speed bus's view of the 1 msec boundaries is called B-Frames.





**Figure 52-11. Relationship of Periodic Schedule Frame Boundaries to Bus Frame Boundaries**

H-Frame boundaries for the host controller correspond to increments of FRINDEX[13:3]. Micro-frame numbers for the H-Frame are tracked by FRINDEX[2:0]. B-Frame boundaries are visible on the high-speed bus through changes in the SOF token's frame number. Micro-frame numbers on the high-speed bus are only derived from the SOF token's frame number (that is the high-speed bus sees eight SOFs with the same frame number value). H-Frames and B-Frames have the fixed relationship (that is B-Frames lag H-Frames by one micro-frame time) illustrated in the figure above. The host controller's periodic schedule is naturally aligned to H-Frames. Software schedules transactions for full- and low-speed periodic endpoints relative the H-Frames. The result is these transactions execute on the high-speed bus at exactly the right time for the USB 2.0 Hub periodic pipeline. As described in [USB Frame Index \(USBC<sub>n</sub>\\_FRINDEX\)](#), the SOF Value can be implemented as a shadow register (in this example, called SOFV), which lags the FRINDEX register bits [13:3] by one micro-frame count. This lag behavior can be accomplished by incrementing FRINDEX[13:3] based on carry-out on the 7 to 0 increment of FRINDEX[2:0] and incrementing SOFV based on the transition of 0 to 1 of FRINDEX[2:0].

Software is allowed to write to FRINDEX. [USB Frame Index \(USBC<sub>n</sub>\\_FRINDEX\)](#) provides the requirements that software should adhere when writing a new value in FRINDEX.

The table below illustrates the required relationship between the value of FRINDEX and the value of SOFV.

**Table 52-39. Operation of FRINDEX and SOFV (SOF Value Register)**

Current	Next
---------	------

*Table continues on the next page...*

**Table 52-39. Operation of FRINDEX and SOFV (SOF Value Register) (continued)**

FRINDEX[F]	SOFV	FRINDEX[mF]	FRINDEX[F]	SOFV	FRINDEX[mF]
N	N	111b	N+1	N	000b
N+1	N	000b	N+1	N+1	001b
N+1	N+1	001b	N+1	N+1	010b
N+1	N+1	010b	N+1	N+1	011b
N+1	N+1	011b	N+1	N+1	100b
N+1	N+1	100b	N+1	N+1	101b
N+1	N+1	101b	N+1	N+1	110b
N+1	N+1	110b	N+1	N+1	111b

**NOTE**

Where [F] = [13:3]; [μF] = [2:0]

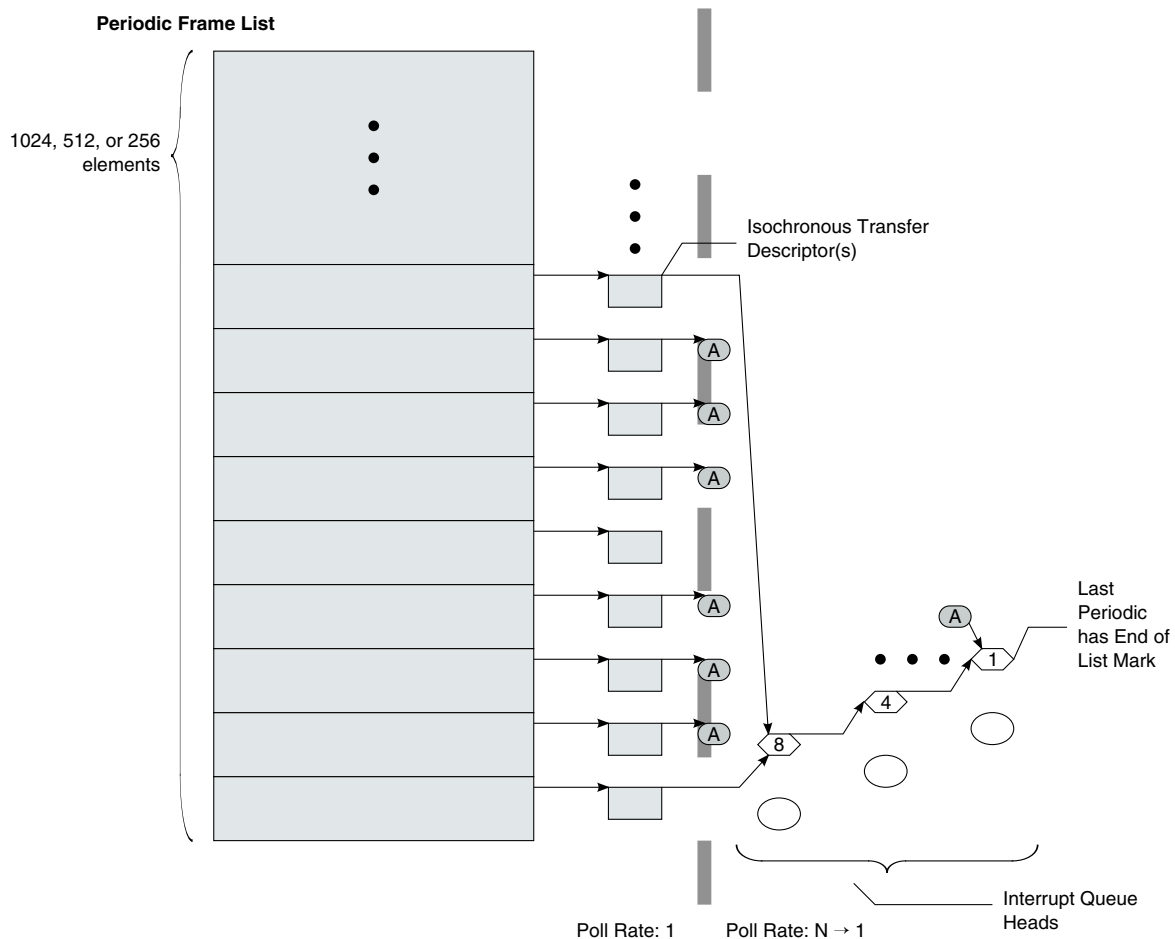
**52.4.3.6 Periodic Schedule**

The periodic schedule traversal is enabled or disabled through the Periodic Schedule Enable bit in the USB\_USBCMD register.

If the Periodic Schedule Enable bit is set to zero, then the host controller simply does not try to access the periodic frame list through the USB\_PERIODICLISTBASE register. Likewise, when the Periodic Schedule Enable bit is one, then the host controller does use the USB\_PERIODICLISTBASE register to traverse the periodic schedule. The host controller will not react to modifications to the Periodic Schedule Enable immediately. In order to eliminate conflicts with split transactions, the host controller evaluates the Periodic Schedule Enable bit only when FRINDEX[2:0] is zero. System software must not disable the periodic schedule if the schedule contains an active split transaction work item that spans the 000b micro-frame. These work items must be removed from the schedule before the Periodic Schedule Enable bit is written to zero. The Periodic Schedule Status bit in the USB\_USBSTS register indicates status of the periodic schedule. System software enables (or disables) the periodic schedule by writing one (or zero) to the Periodic Schedule Enable bit in the USB\_USBCMD register. Software then can poll the Periodic Schedule Status bit to determine when the periodic schedule has made the desired transition. Software must not modify the Periodic Schedule Enable bit unless the value of the Periodic Schedule Enable bit equals that of the Periodic Schedule Status bit.

The periodic schedule is used to manage all isochronous and interrupt transfer streams. The base of the periodic schedule is the periodic frame list. Software links schedule data structures to the periodic frame list to produce a graph of scheduled data structures. The graph represents an appropriate sequence of transactions on the

The following figure illustrates isochronous transfers (using iTDs and siTDs) with a period of one are linked directly to the periodic frame list. Interrupt transfers (are managed with queue heads) and isochronous streams with periods other than one are linked following the period-one iTD/siTDs. Interrupt queue heads are linked into the frame list ordered by poll rate. Longer poll rates are linked first (for example, closest to the periodic frame list), followed by shorter poll rates, with queue heads with a poll rate of one, on the very end.



**Figure 52-12. Example Periodic Schedule**

### 52.4.3.7 Managing Isochronous Transfers Using iTDs

The structure of an iTD is presented in [Isochronous \(High-Speed\) Transfer Descriptor \(iTD\)](#). The four distinct sections to an iTD:

- The first field is the Next Link Pointer. This field is for schedule linkage purposes only.

- Transaction description array. This area is an eight-element array. Each element represents control and status information for one micro-frame's worth of transactions for a single high-speed isochronous endpoint.
- The buffer page pointer array is a 7-element array of physical memory pointers to data buffers. These are 4 K aligned pointers to physical memory.
- Endpoint capabilities. This area utilizes the unused low-order 12 bits of the buffer page pointer array. The fields in this area are used across all transactions executed for this iTD, including endpoint addressing, transfer direction, maximum packet size and high-bandwidth multiplier.

#### 52.4.3.7.1 Host Controller Operational Model for iTDs

The host controller uses FRINDEX register bits [12:3] to index into the periodic frame list. This means that the host controller visits each frame list element eight consecutive times before incrementing to the next periodic frame list element. Each iTD contains eight transaction descriptions, which map directly to FRINDEX register bits [2:0]. Therefore, each transaction descriptor corresponds to one micro-frame. Each iTD can span 8 micro-frames worth of transactions.

When the host controller fetches an iTD, it uses FRINDEX register bits [2:0] to index into the transaction description array.

If the active bit in the Status field of the indexed transaction description is set to zero, the host controller ignores the iTD and follows the Next pointer to the next schedule data structure.

When the indexed active bit is one, the host controller continues to parse the iTD. It stores the indexed transaction description and the general endpoint information (device address, endpoint number, maximum packet size, and so on.). It also uses the Page Select (PG) field to index the buffer pointer array, storing the selected buffer pointer and the next sequential buffer pointer. For example, if PG field is 0, then the host controller stores Page 0 and Page 1.

The host controller constructs a physical data buffer address by concatenating the current buffer pointer (as selected using the current transaction description's PG field) and the transaction description's Transaction Offset field. The host controller uses the endpoint addressing information and I/O-bit to execute a transaction to the appropriate endpoint. When the transaction is complete, the host controller clears the active bit and writes back any additional status information to the Status field in the currently selected transaction description.

The data buffer associated with the iTD must be virtually contiguous memory. Seven page pointers are provided to support eight high-bandwidth transactions regardless of the starting packet's offset alignment into the first page. A starting buffer pointer (physical

memory address) is constructed by concatenating the page pointer (for example, page 0 pointer) selected by the active transaction descriptions' PG (for example, value: 00B) field with the transaction offset field. As the transaction moves data, the host controller must detect when an increment of the current buffer pointer crosses a page boundary. When this occurs the host controller simply replaces the current buffer pointer's page portion with the next page pointer (for example, page 1 pointer) and continues to move data. The size of each bus transaction is determined by the value in the Maximum Packet Size field. An iTD supports high-bandwidth pipes through the Mult (multiplier) field. When the Mult field is 1, 2, or 3, the host controller executes the specified number of Maximum Packet sized bus transactions for the endpoint in the current micro-frame. In other words, the Mult field represents a transaction count for the endpoint in the current micro-frame. If the Mult field is zero, the operation of the host controller is undefined. The transfer description is used to service all transactions indicated by the Mult field.

For OUT transfers, the value of the Transaction X Length field represents the total bytes to be sent during the micro-frame. The Mult field must be set by software to be consistent with Transaction X Length and Maximum Packet Size. The host controller sends the bytes in Maximum Packet Size'd portions. After each transaction, the host controller decrements its local copy of Transaction X Length by Maximum Packet Size. The number of bytes the host controller sends is always Maximum Packet Size or Transaction X Length, whichever is less. The host controller advances the transfer state in the transfer description, updates the appropriate record in the iTD and moves to the next schedule data structure. The maximum sized transaction supported is 3 x 1024 bytes.

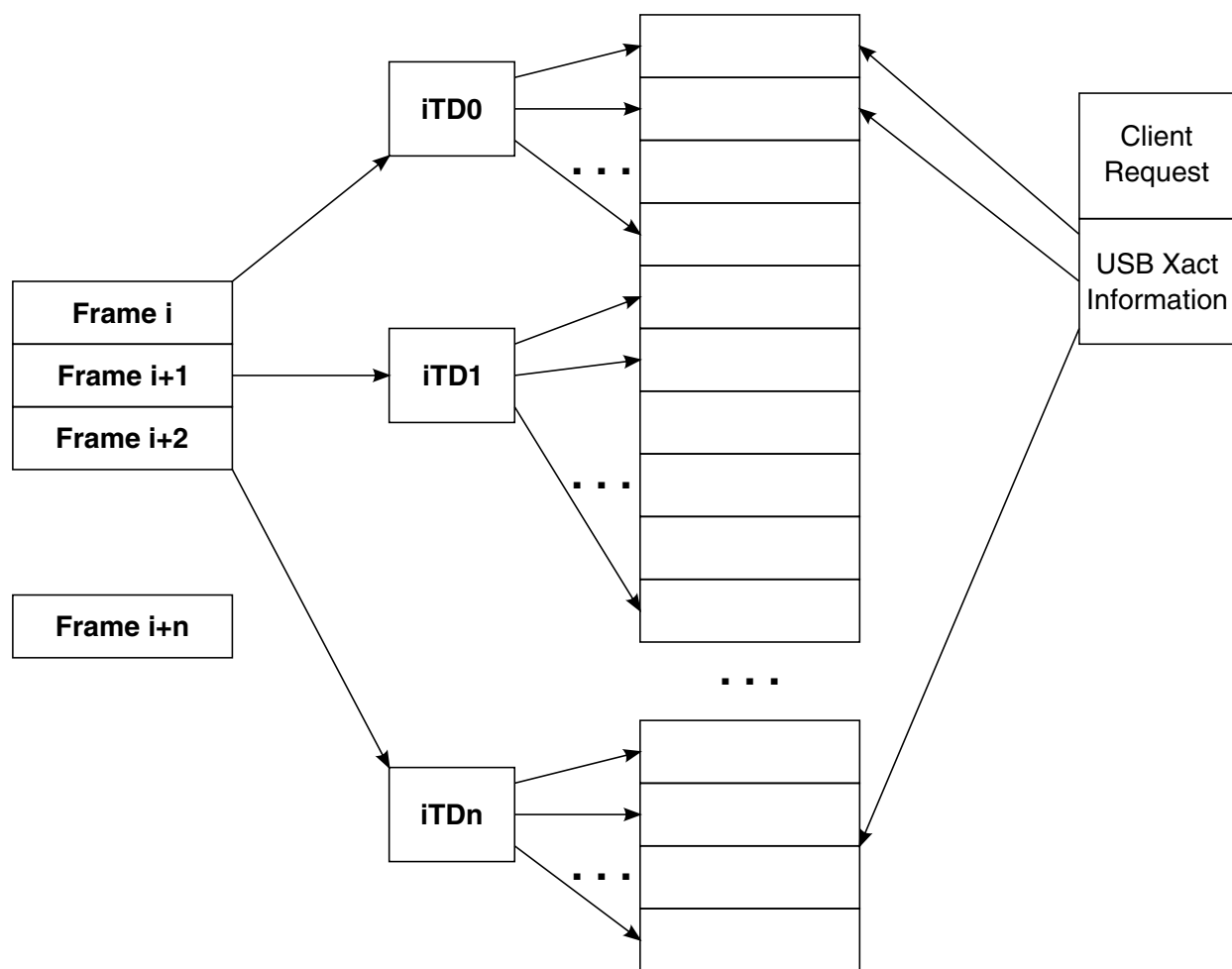
For IN transfers, the host controller issues Mult transactions. It is assumed that software has properly initialized the iTD to accommodate all of the possible data. During each IN transaction, the host controller must use Maximum Packet Size to detect packet babble errors. The host controller keeps the sum of bytes received in the Transaction X Length field. After all transactions for the endpoint have completed for the micro-frame, Transaction X Length contains the total bytes received. If the final value of Transaction X Length is less than the value of Maximum Packet Size, then less data than was allowed for was received from the associated endpoint. This short packet condition does not set the USBINT bit in the USB\_USBSTS register to one. The host controller will not detect this condition. If the device sends more than Transaction X Length or Maximum Packet Size bytes (whichever is less), then the host controller sets the Babble Detected bit to one and set the Active bit to zero. Note, that the host controller is not required to update the iTD field Transaction X Length in this error scenario. If the Mult field is greater than one, then the host controller automatically executes the value of Mult transactions. The host controller will not execute all Mult transactions if:

- The endpoint is an OUT and Transaction X Length goes to zero before all the Mult transactions have executed (ran out of data), or
- The endpoint is an IN and the endpoint delivers a short packet, or an error occurs on a transaction before Mult transactions have been executed. The end of micro-frame may occur before all of the transaction opportunities have been executed. When this happens, the transfer state of the transfer description is advanced to reflect the progress that was made, the result written back to the iTD and the host controller proceeds to processing the next micro-frame. Refer to Appendix D for a table summary of the host controller required behavior for all the high-bandwidth transaction cases.

#### 52.4.3.7.2 Software Operational Model for iTDs

A client buffer request to an isochronous endpoint may span 1 to N micro-frames. When N is larger than one, system software may have to use multiple iTDs to read or write data with the buffer (if N is larger than eight, it must use more than one iTD).

The following figure illustrates the simple model of how a client buffer is mapped by system software to the periodic schedule (that is the periodic frame list and a set of iTDs). On the right is the client description of its request. The description includes a buffer base address plus additional annotations to identify which portions of the buffer should be used with each bus transaction. In the middle is the iTD data structures used by the system software to service the client request. Each iTD can be initialized to service up to 24 transactions, organized into eight groups of up to three transactions each. Each group maps to one micro-frame's worth of transactions. The EHCI controller does not provide per-transaction results within a micro-frame. It treats the per-micro-frame transactions as a single logical transfer. On the left is the host controller's frame list. System software establishes references from the appropriate locations in the frame list to each of the appropriate iTDs. If the buffer is large, then system software can use a small set of iTDs to service the entire buffer. System software can activate the transaction description records (contained in each iTD) in any pattern required for the particular data stream.



**Figure 52-13. Example Association of iTDs to Client Request Buffer**

As noted above, the client request includes a pointer to the base of the buffer and offsets into the buffer to annotate which buffer sections are to be used on each bus transaction that occurs on this endpoint. System software must initialize each transaction description in an iTD to ensure it uses the correct portion of the client buffer. For example, for each transaction description, the PG field is set to index the correct physical buffer page pointer and the Transaction Offset field is set relative to the correct buffer pointer page (for example, the same one referenced by the PG field). When the host controller executes a transaction it selects a transaction description record based on FRINDEX[2:0]. It then uses the current Page Buffer Pointer (as selected by the PG field) and concatenates to the transaction offset field. The result is a starting buffer address for the transaction. As the host controller moves data for the transaction, it must watch for a page wrap condition and properly advance to the next available Page Buffer Pointer. System software must not use the Page 6 buffer pointer in a transaction description where the length of the transfer wraps a page boundary. Doing so yields undefined behavior. The host controller hardware is not required to 'alias' the page selector to Page zero. USB 2.0 isochronous

endpoints can specify a period greater than one. Software can achieve the appropriate scheduling by linking iTDs into the appropriate frames (relative to the frame list) and by setting appropriate transaction description elements active bits to one.

#### 52.4.3.7.2.1 Periodic scheduling threshold

The Isochronous Scheduling Threshold field in the USB\_HCCPARAMS capability register is an indicator to system software as to how the host controller pre-fetches and effectively caches schedule data structures.

It is used by system software when adding isochronous work items to the periodic schedule. The value of this field indicates to system software the minimum distance it can update isochronous data (relative to the current location of the host controller execution in the periodic list) and still have the host controller process them.

The iTD and siTD data structures each describe 8 micro-frames worth of transactions. The host controller is allowed to cache one (or more) of these data structures in order to reduce memory traffic. Three basic caching models that account for the fact the isochronous data structures span 8 micro-frames. The three caching models are: no caching, micro-frame caching and frame caching.

When software is adding new isochronous transactions to the schedule, it always performs a read of the USB\_FRINDEX register to determine the current frame and micro-frame the host controller is currently executing. Of course, there is no information about where in the micro-frame the host controller is, so a constant uncertainty-factor of one micro-frame has to be assumed. Combining the knowledge of where the host controller is executing with the knowledge of the caching model allows the definition of simple algorithms for how closely software can reliably work to the executing host controller.

No caching is indicated with a value of zero in the Isochronous Scheduling Threshold field. The host controller may pre-fetch data structures during a periodic schedule traversal (per micro-frame) but always dumps any accumulated schedule state at the end of the micro-frame. At the appropriate time relative to the beginning of every micro-frame, the host controller always begins schedule traversal from the frame list. Software can use the value of the USB\_FRINDEX register (plus the constant 1 uncertainty-factor) to determine the approximate position of the executing host controller. When no caching is selected, software can add an isochronous transaction as near as 2 micro-frames in front of the current executing position of the host controller.

Frame caching is indicated with a non-zero value in bit [7] of the Isochronous Scheduling Threshold field. In the frame-caching model, system software assumes that the host controller caches one (or more) isochronous data structures for an entire frame (8 micro-frames). Software uses the value of the USB\_FRINDEX register (plus the constant 1



uncertainty) to determine the current micro-frame/frame (assume modulo 8 arithmetic in adding the constant 1 to the micro-frame number). For any current frame N, if the current micro-frame is 0 to 6, then software can safely add isochronous transactions to Frame N + 1. If the current micro-frame is 7, then software can add isochronous transactions to Frame N + 2.

Micro-frame caching is indicated with a non-zero value in the least-significant 3 bits of the Isochronous Scheduling Threshold field. System software assumes the host controller caches one or more periodic data structures for the number of micro-frames indicated in the Isochronous Scheduling Threshold field. For example, if the count value were 2, then the host controller keeps a window of 2 micro-frames worth of state (current micro-frame, plus the next) on-chip. On each micro-frame boundary, the host controller releases the current micro-frame state and begins accumulating the next micro-frame state.

### 52.4.3.8 Asynchronous Schedule

The Asynchronous schedule traversal is enabled or disabled through the Asynchronous Schedule Enable bit in the USB\_USBCMD register.

If the Asynchronous Schedule Enable bit is set to zero, then the host controller simply does not try to access the asynchronous schedule through the USB\_ASYNC\_LISTADDR register. Likewise, when the Asynchronous Schedule Enable bit is one, then the host controller does use the USB\_ASYNC\_LISTADDR register to traverse the asynchronous schedule. Modifications to the Asynchronous Schedule Enable bit are not necessarily immediate. Rather the new value of the bit is taken into consideration the next time the host controller needs to use the value of the USB\_ASYNC\_LISTADDR register to get the next queue head.

The Asynchronous Schedule Status bit in the USB\_USBSTS register indicates status of the asynchronous schedule. System software enables (or disables) the asynchronous schedule by writing one (or zero) to the Asynchronous Schedule Enable bit in the USB\_USBCMD register. Software then can poll the Asynchronous Schedule Status bit to determine when the asynchronous schedule has made the desired transition. Software must not modify the Asynchronous Schedule Enable bit unless the value of the Asynchronous Schedule Enable bit equals that of the Asynchronous Schedule Status bit.

The asynchronous schedule is used to manage all Control and Bulk transfers. Control and Bulk transfers are managed using queue head data structures. The asynchronous schedule is based at the USB\_ASYNC\_LISTADDR register. The default value of the USB\_ASYNC\_LISTADDR register after reset is undefined and the schedule is disabled when the Asynchronous Schedule Enable bit is zero.

Software may only write this register with defined results when the schedule is disabled. For example, Asynchronous Schedule Enable bit in the USB\_USBCMD and the Asynchronous Schedule Status bit in the USB\_USBSTS register are zero. System software enables execution from the asynchronous schedule by writing a valid memory address (of a queue head) into this register. Then software enables the asynchronous schedule by setting the Asynchronous Schedule Enable bit to one. The asynchronous schedule is actually enabled when the Asynchronous Schedule Status bit is one.

When the host controller begins servicing the asynchronous schedule, it begins by using the value of the USB\_ASYNC\_LIST\_ADDR register. It reads the first referenced data structure and begins executing transactions and traversing the linked list as appropriate. When the host controller completes processing the asynchronous schedule, it retains the value of the last accessed queue head's horizontal pointer in the USB\_ASYNC\_LIST\_ADDR register. Next time the asynchronous schedule is accessed, this is the first data structure that is serviced. This provides round-robin fairness for processing the asynchronous schedule.

A host controller completes processing the asynchronous schedule when one of the following events occur:

- The end of a micro-frame occurs.
- The host controller detects an empty list condition (see [Empty Asynchronous Schedule Detection](#) )
- The schedule has been disabled through the Asynchronous Schedule Enable bit in the USB\_USBCMD register.

The queue heads in the asynchronous list are linked into a simple circular list as shown in [Figure 52-8](#). Queue head data structures are the only valid data structures that may be linked into the asynchronous schedule. An isochronous transfer descriptor (iTDP or siTD) in the asynchronous schedule yields undefined results.

The maximum packet size field in a queue head is sized to accommodate the use of this data structure for all non-isochronous transfer types. The USB Specification, Revision 2.0 specifies the maximum packet sizes for all transfer types and transfer speeds. System software should always parameterize the queue head data structures according to the core specification requirements.

#### 52.4.3.8.1 Adding Queue Heads to Asynchronous Schedule

This is a software requirement section.

There are two independent events for adding queue heads to the asynchronous schedule. The first is the initial activation of the asynchronous list. The second is inserting a new queue head into an activated asynchronous list.

Activation of the list is simple. System software writes the physical memory address of a queue head into the USB\_ASYNC\_LIST\_ADDR register, then enables the list by setting the Asynchronous Schedule Enable bit in the USB\_USBCMD register to one.

When inserting a queue head into an active list, software must ensure that the schedule is always coherent from the host controllers' point of view. This means that the system software must ensure that all queue head pointer fields are valid. For example, qTD pointers have T-Bits set to one or reference valid qTDs and the Horizontal Pointer references a valid queue head data structure. The following algorithm represents the functional requirements:

```
InsertQueueHead (pQHeadCurrent, pQueueHeadNew)
--
-- Requirement: all inputs must be properly initialized.
--
-- pQHeadCurrent is a pointer to a queue head that is
-- already in the active list
-- pQHeadNew is a pointer to the queue head to be added
--
-- This algorithm links a new queue head into a existing
-- list
--
pQueueHeadNew.HorizontalPointer = pQueueHeadCurrent.HorizontalPointer
pQueueHeadCurrent.HorizontalPointer = physicalAddressOf (pQueueHeadNew)
End InsertQueueHead
```

#### 52.4.3.8.2 Removing Queue Heads from Asynchronous Schedule

This is a software requirement section.

There are two independent events for removing queue heads from the asynchronous schedule. The first is shutting down (deactivating) the asynchronous list. The second is extracting a single queue head from an activated list.

Software deactivates the asynchronous schedule by setting the Asynchronous Schedule Enable bit in the USB\_USBCMD register to zero. Software can determine when the list is idle when the Asynchronous Schedule Status bit in the USB\_USBSTS register is zero. The normal mode of operation is that software removes queue heads from the asynchronous schedule without shutting it down. Software must not remove an active queue head from the schedule. Software should first deactivate all active qTDs, wait for the queue head to go inactive, then remove the queue head from the asynchronous list. Software removes a queue head from the asynchronous list through the following algorithm. As illustrated, the unlinking is quite easy. Software merely must ensure all of the link pointers reachable by the host controller are kept consistent.

```
UnlinkQueueHead (pQHeadPrevious, pQueueHeadToUnlink, pQHeadNext)
--
-- Requirement: all inputs must be properly initialized.
--
-- pQHeadPrevious is a pointer to a queue head that
-- references the queue head to remove
-- pQHeadToUnlink is a pointer to the queue head to be
```

```
-- removed
-- pQHeadNext is a pointer to a queue head still in the
-- schedule. Software provides this pointer with the
-- following strict rules:
--     if the host software is one queue head, then
--     pQHeadNext must be the same as
--     QueueheadToUnlink.HorizontalPointer. If the host
--     software is unlinking a consecutive series of
--     queue heads, QHeadNext must be set by software to
--     the queue head remaining in the schedule.
--
-- This algorithm unlinks a queue head from a circular list
--
pQueueHeadPrevious.HorizontalPointer = pQueueHeadToUnlink.HorizontalPointer
pQueueHeadToUnlink.HorizontalPointer = pQHeadNext
```

End UnlinkQueueHead

If software removes the queue head with the H-bit set to one, it must select another queue head still linked into the schedule and set its H-bit to one. This should be completed before removing the queue head. The requirement is that software keep one queue head in the asynchronous schedule, with its H-bit set to one. At the point software has removed one or more queue heads from the asynchronous schedule, it is unknown whether the host controller has a cached pointer to them. Similarly, it is unknown how long the host controller might retain the cached information, as it is implementation dependent and may be affected by the actual dynamics of the schedule load. Therefore, once software has removed a queue head from the asynchronous list, it must retain the coherency of the queue head (link pointers, and so on). It cannot disturb the removed queue heads until it knows that the host controller does not have a local copy of a pointer to any of the removed data structures.

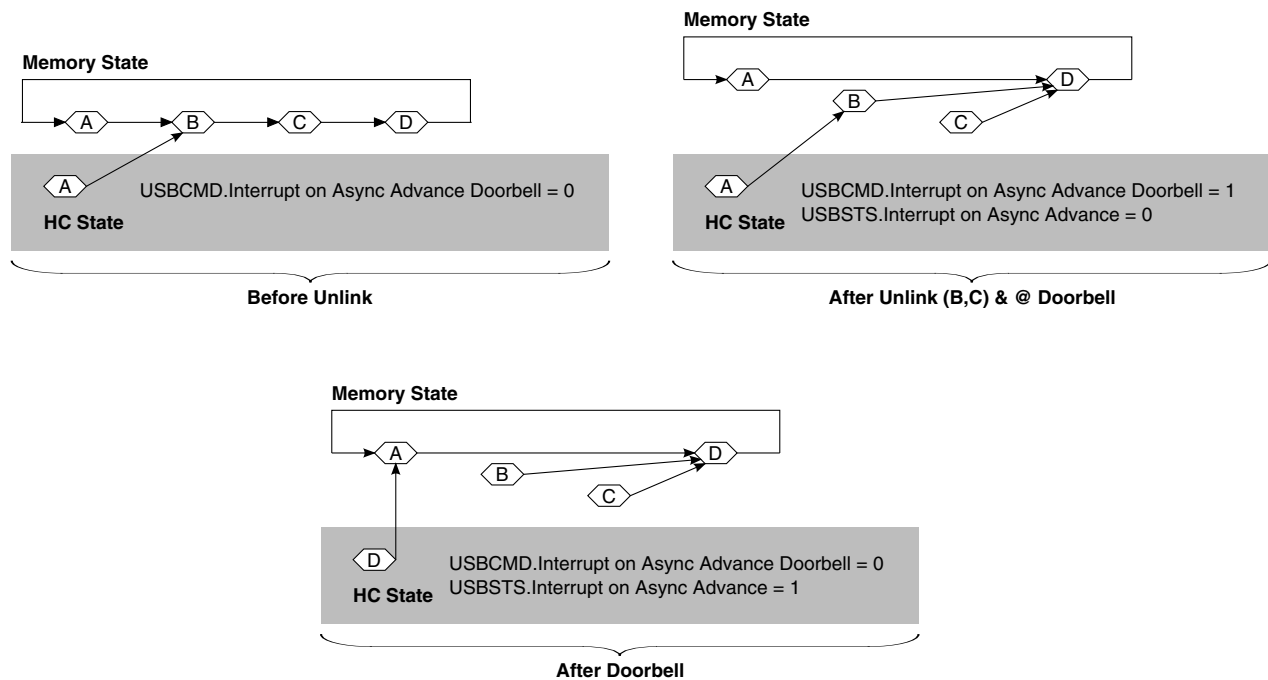
The method software uses to determine when it is safe to modify a removed queue head is to handshake with the host controller. The handshake mechanism allows software to remove items from the asynchronous schedule, then execute a simple, lightweight handshake that is used by software as a key that it can free (or reuse) the memory associated the data structures it has removed from the asynchronous schedule.

The handshake is implemented with three bits in the host controller. The first bit is a command bit (Interrupt on Async Advance Doorbell bit in the USB\_USBCMD register) that allows software to inform the host controller that something has been removed from its asynchronous schedule. The second bit is a status bit (Interrupt on Async Advance bit in the USB\_USBSTS register) that the host controller sets after it has released all on-chip state that may potentially reference one of the data structures just removed. When the host controller sets this status bit to one, it also sets the command bit to zero. The third bit is an interrupt enable (Interrupt on Async Advance bit in the USB\_USBINTR register) that is matched with the status bit. If the status bit is one and the interrupt enable bit is one, then the host controller asserts a hardware interrupt.

The figure below illustrates a general example. In this example, consecutive queue heads (B and C) are unlinked from the schedule using the algorithm above. Before the unlink operation, the host controller has a copy of queue head A.

The unlink algorithm requires that as software unlinks each queue head, the unlinked queue head is loaded with the address of a queue head that remains in the asynchronous schedule.

When the host controller observes that doorbell bit being set to one, it makes a note of the local reachable schedule information. In this example, the local reachable schedule information includes both queue heads (A and B). It is sufficient that the host controller can set the status bit (and clear the doorbell bit) as soon as it has traversed beyond current reachable schedule information (that is traversed beyond queue head (B) in this example). The following figure illustrates the generic queue head unlink scenario.



**Figure 52-14. Generic Queue Head Unlink Scenario**

Alternatively, a host controller implementation is allowed to traverse the entire asynchronous schedule list (for example, observed the head of the queue (twice)) before setting the Advance on Async status bit to one.

Software may re-use the memory associated with the removed queue heads after it observes the Interrupt on Async Advance status bit is set to one, following assertion of the doorbell. Software should acknowledge the Interrupt on Async Advance status as indicated in the `USB_USBSTS` register, before using the doorbell handshake again.

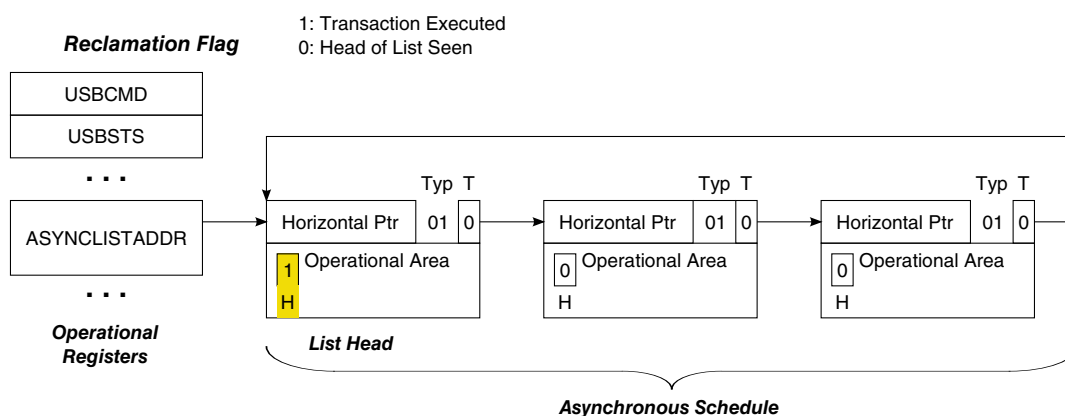
### 52.4.3.8.3 Empty Asynchronous Schedule Detection

The Enhanced Host Controller Interface uses two bits to detect when the asynchronous schedule is empty.

The queue head data structure (see [Table 52-25](#)) defines an *H-bit* in the queue head, which allows software to mark a queue head as being the *head* of the reclaim list. The Enhanced Host Controller Interface also keeps a 1-bit flag in the USB\_USBSTS register (*Reclamation*) that is set to zero when the Enhanced Interface Host Controller observes a queue head with the H-bit set to one. The reclamation flag in the status register is set to one when any USB transaction from the asynchronous schedule is executed (or whenever the asynchronous schedule starts, see [Asynchronous schedule traversal: Start Event](#)).

If the Enhanced Host Controller Interface ever encounters an *H-bit* of one and a *Reclamation* bit of zero, the EHCI controller simply stops traversal of the asynchronous schedule.

An example illustrating the H-bit in a schedule is shown in the following figure.



**Figure 52-15. Asynchronous Schedule List w/Annotation to Mark Head of List**

Software must ensure there is at most one queue head with the *H-bit* set to one, and that it is always coherent with respect to the schedule.

#### 52.4.3.8.4 Restarting Asynchronous Schedule Before EOF

There are many situations where the host controller will detect an empty list *long* before the end of the micro-frame.

It is important to remember that under many circumstances the schedule traversal has stopped due to Nak/Nyet responses from all endpoints.

An example of particular interest is when a start-split for a bulk endpoint occurs early in the micro-frame. Given the EHCI simple traversal rules, the complete-split for that transaction may Nak/Nyet out very quickly. If it is the only item in the schedule, then the host controller ceases traversal of the Asynchronous schedule very early in the micro-frame. In order to provide reasonable service to this endpoint, the host controller should issue the complete-split before the end of the current micro-frame, instead of waiting

until the next micro-frame. When the reason for host controller idling asynchronous schedule traversal is because of empty list detection, it is mandatory the host controller implement a 'waking' method to resume traversal of the asynchronous schedule. An example method is described below.

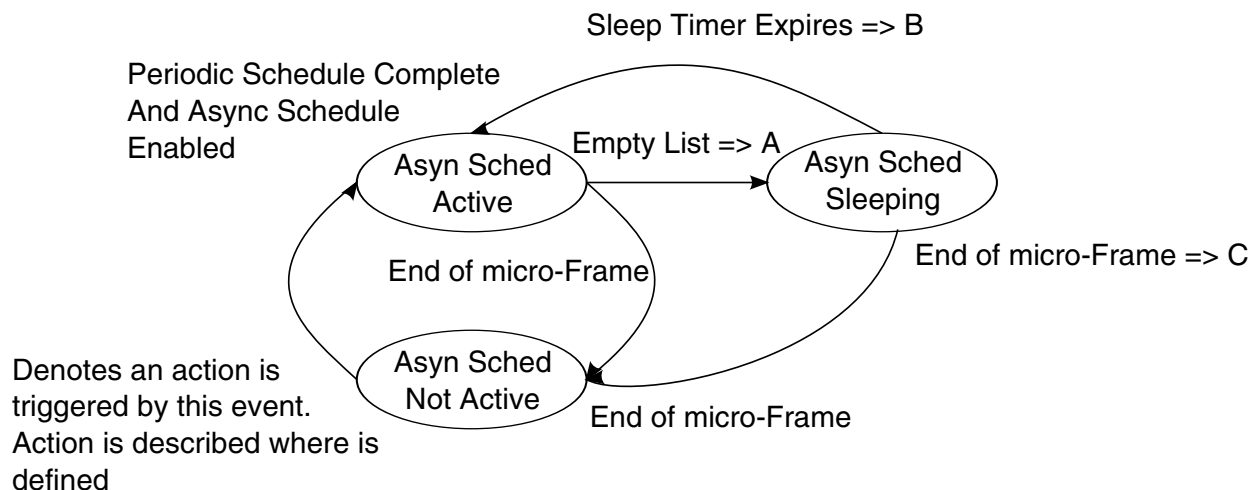
#### 52.4.3.8.4.1 Example Method for Restarting Asynchronous Schedule Traversal

The reason for idling the host controller when the list is empty is to keep the host controller from unnecessarily occupying too much memory bandwidth. The question is: *how long should the host controller stay idle before restarting?*

The answer in this example is based on deriving a manifest constant, which is the amount of time the host controller will stay idle before restarting traversal. In this example, the manifest constant is called *AsyncSchedSleepTime*, and has a value of 10  $\mu$ sec. The value is derived based on the analysis in [Example Derivation for AsyncSchedSleepTime](#). The traversal algorithm is simple:

- Traverse the Asynchronous schedule until the either an End-Of-micro-Frame event occurs, or an empty list is detected. If the event is an End-of-micro-Frame, go attempt to traverse the Periodic schedule. If the event is an empty list, then set a sleep timer and go to a *schedule sleep* state.
- When the sleep timer expires, set working context to the Asynchronous Schedule start condition and go to *schedule active* state. The start context allows the HC to reload *Nakcnt* fields, and so on. So the HC has a chance to run for more than one iteration through the schedule.

This process simply repeats itself each micro-frame. The figure below illustrates a sample state machine to manage the active and sleep states of the Asynchronous Schedule traversal policy. There are three states: Actively traversing the Asynchronous schedule, Sleeping, and Not Active. The last two are similar in terms of interaction with the Asynchronous schedule, but the Not Active state means that the host controller is busy with the Periodic schedule or the Asynchronous schedule is not enabled. The Sleeping state is specifically a special state where the host controller is just waiting for a period of time before resuming execution of the Asynchronous schedule.



**Figure 52-16. Example State Machine for Managing Asynchronous Schedule Traversal**

The actions referred to in the figure above are defined in the following table.

**Table 52-40. Asynchronous Schedule SM Transition Actions**

Action	Action Description Label
A	On detection of the empty list, the host controller sets the <i>AsynchronousTraversalSleepTimer</i> to <i>AsynSchedSleepTime</i> .
B	When the <i>AsynchronousTraversalSleepTimer</i> expires, the host controller sets the <i>Reclamation</i> bit in the USBSTS register to one and moves the Nak Counter reload state machine to WaitForListHead (see <a href="#">Nak Count Reload Control</a> ).
C	The host controller cancels the sleep timer ( <i>AsynchronousTraversalSleepTimer</i> ).

**52.4.3.8.4.2 Async Sched Not Active**

This is the initial state of the traversal state machine after a host controller reset. The traversal state machine does not leave this state when the *Asynchronous Schedule Enable* bit in the USB\_USBCMD register is zero.

This state is entered from Async Sched Active or Async Sched Sleeping states when the end-of-micro-frame event is detected.

**52.4.3.8.4.3 Async Sched Active**

This state is entered from the Async Sched Not Active state when the periodic schedule is not active. It is also entered from the Async Sched Sleeping states when the *AsynchrhonousTraversalSleepTimer* expires. On every transition into this state, the host controller sets the *Reclamation* bit in the USB\_USBSTS register to one.



While in this state, the host controller continually traverses the asynchronous schedule until either the end of micro-frame or an empty list condition is detected.

#### 52.4.3.8.4.4 Async Sched Sleeping

The state is entered from the Async Sched Active state when a schedule empty condition is detected. On entry to this state, the host controller sets the *AsynchronousTraversalSleepTimer* to *AsyncSchedSleepTime*.

#### 52.4.3.8.4.5 Example Derivation for AsyncSchedSleepTime

The derivation is based on analysis of what work the host controller could be doing next.

It assumes the host controller does not keep any state about what work is possibly pending in the asynchronous schedule. The schedule could contain any mix of the possible combinations of high- full- or low-speed control and bulk requests.

The table below summarizes some of the typical 'next transactions' that could be in the schedule, and the amount of time (for example *footprint*, or *wall clock*) the transaction takes to complete.

**Table 52-41. Typical Low-/Full-speed Transaction Times**

Transaction Attributes		Footprint (time)	Description
Speed	HS	11.9 ms	Maximum foot print for a worst-case, full-sized bulk data transaction.
Size	512	9.45 ms	Maximum footprint for an approximate best-case, full-sized bulk data transaction.
Type	Bulk		
Speed	FS	~50 ms	Approximate typical for full-sized bulk data. An 8-byte low-speed is about 2x, or between 90 and 100 ms.
Size	64		
Type	Bulk		
Speed	FS	~12 ms	Approximate typical for 8-byte bulk/control (that is setup)
Size	8		
Type	Cntrl		

A *AsyncSchedSleepTime* value of 10  $\mu$ s provides a reasonable relaxation of the system memory load and still provides a good level of service for the various transfer types and payload sizes. For example, say we detect an empty list after issuing a start-split for a 64-byte full-speed bulk request. Assuming this is the only thing in the list, the host controller gets the results of the full-speed transaction from the hub during the fifth complete-split request. If the full-speed transaction was an IN and it nak'd, the 10  $\mu$ s sleep period would allow the host controller to get the NAK results on the first complete-split.

#### 52.4.3.8.5 Asynchronous schedule traversal: *Start Event*

Once the HC has *idled* itself through the empty schedule detection (Section 0), it will naturally *activate* and begin processing from the Periodic Schedule at the beginning of each micro-frame.

In addition, it may have idled itself early in a micro-frame. When this occurs (idles early in the micro-frame) the HC must occasionally *re-activate* during the micro-frame and traverse the asynchronous schedule to determine whether any progress can be made. The requirements and method for this restart are described in [Restarting Asynchronous Schedule Before EOF](#). Asynchronous schedule *Start Events* are defined to be:

- Whenever the host controller transitions from the periodic schedule to the asynchronous schedule. If the periodic schedule is disabled and the asynchronous schedule is enabled, then the beginning of the micro-frame is equivalent to the transition from the periodic schedule, or
- The asynchronous schedule traversal restarts from a sleeping state (see [Restarting Asynchronous Schedule Before EOF](#)).

#### 52.4.3.8.6 Reclamation Status Bit (USBSTS Register)

The operation of the empty asynchronous schedule detection feature (see [Empty Asynchronous Schedule Detection](#)) depends on the proper management of the *Reclamation* bit in the USB\_USBSTS register.

The host controller tests for an empty schedule just after it fetches a new queue head while traversing the asynchronous schedule (see [Fetch Queue Head](#)).

It is required that the host controller sets the *Reclamation* bit to one whenever an asynchronous schedule traversal *Start Event*, as documented in [Asynchronous schedule traversal: Start Event](#), occurs. The *Reclamation* bit is also set to one whenever the host controller executes a transaction while traversing the asynchronous schedule (see [Execute Transaction](#)). The host controller sets the *Reclamation* bit to zero whenever it finds a queue head with its *H-bit* set to one. Software should only set a queue head's *H-bit* if the queue head is in the asynchronous schedule. If software sets the *H-bit* in an interrupt queue head to one, the resulting behavior is undefined. The host controller may set the *Reclamation* bit to zero when executing from the periodic schedule.

#### 52.4.3.9 Operational Model for Nak Counter

This section describes the operational model for the *NakCnt* field defined in a queue head.

See [Queue Head Initialization](#) for more information. Software should not use this feature for interrupt queue heads. This rule is not required to be enforced by the host controller.

USB protocol has built-in flow control through the Nak response by a device. There are several scenarios, beyond the Ping feature, where an endpoint may naturally Nak or Nyet the majority of the time. An example is the host controller management of the split transaction protocol for control and bulk endpoints. All bulk endpoints (High- or Full-speed) are serviced through the same asynchronous schedule. The time between the *Start-split* transaction and the first *Complete-split* transaction could be very short (that is like when the endpoint is the only one in the asynchronous schedule). The hub NYETs (effectively Naks) the *Complete-split* transaction until the classic transaction is complete. This could result in the host controller thrashing memory, repeatedly fetching the queue head and executing the transaction to the Hub, which does not complete until after the transaction on the classic bus completes.

The two component fields in a queue head to support the throttling feature: a counter field (*NakCnt*), and a counter reload field (*RL*). *NakCnt* is used by the host controller as one of the criteria to determine whether or not to execute a transaction to the endpoint. The two operational modes associated with this counter:

- Not Used- This mode is set when the *RL* field is zero. The host controller ignores the *NakCnt* field for any execution of transactions through a queue head with an *RL* field of zero. Software must use this selection for interrupt endpoints.
- Nak Throttle Mode- This mode is selected when the *RL* field is non-zero. In this mode, the value in the *NakCnt* field represents the maximum number of Nak or Nyet responses the host controller tolerates on each endpoint. In this mode, the HC decrements the *NakCnt* field based on the token/handshake criteria listed in the table below. The host controller must reload *NakCnt* when the endpoint successfully moves data (for example, policy to reward device for moving data).

The following table describes the *NakCnt* field adjustment rules.

**Table 52-42. NakCnt Field Adjustment Rules**

Token	Handshake	
	Handshake NAK	NYET
IN/PING	decrement <i>NakCnt</i>	N/A (protocol error)
OUT	decrement <i>NakCnt</i>	No Action <sup>1</sup> Start
Split	decrement <i>NakCnt</i>	N/A (protocol error)
Complete Split	No Action	Decrement <i>NakCnt</i>

1. Recommended behavior on this response is to reload *NakCnt*

In summary, system software enables the counter by setting the reload field (*RL*) to a non-zero value. The host controller may execute a transaction if *NakCnt* is non-zero. The host controller does not execute a transaction if *NakCnt* is zero. The reload mechanism is described in detail in [Nak Count Reload Control](#).

### NOTE

When all queue heads in the Asynchronous Schedule either exhausts all transfers or all *NakCnt*'s go to zero, then the host controller detects an empty Asynchronous Schedule and idle schedule traversal (see [Empty Asynchronous Schedule Detection](#)).

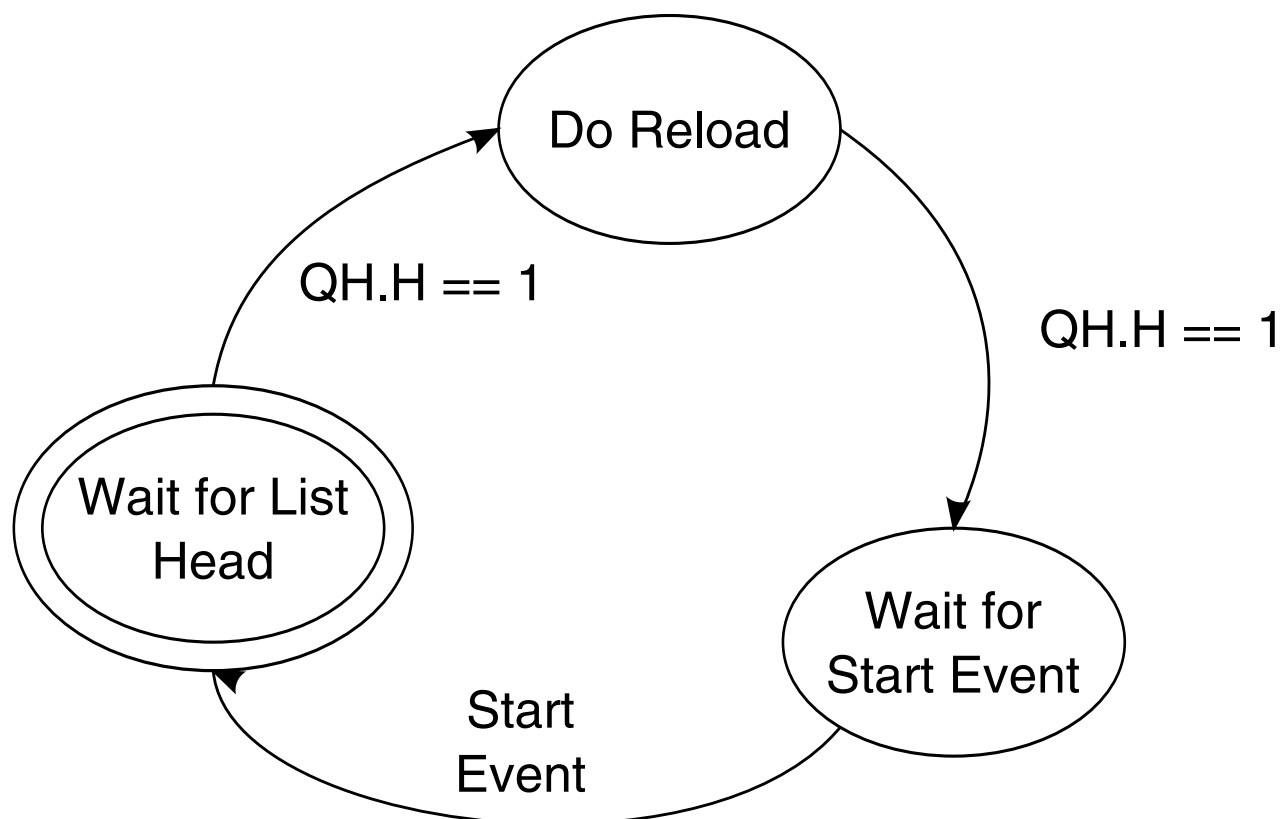
Any time the host controller begins a new traversal of the Asynchronous Schedule, a *Start Event* is assumed, see [Asynchronous schedule traversal: Start Event](#). Every time a Start-Event occurs, the Nak Count reload procedure is enabled.

#### 52.4.3.9.1 Nak Count Reload Control

When the host controller reaches the *Execute Transaction* state for a queue head (meaning that it has an active operational state), it checks to determine whether the *NakCnt* field should be reloaded from *RL* (see [Execute Transaction](#)). If the answer is yes, then *RL* is copied into *NakCnt*. After the reload or if the reload is not active, the host controller evaluates whether to execute the transaction.

The host controller must reload nak counters (*NakCnt* see [Table 52-25](#)) in queue heads during the first pass through the reclamation list after an asynchronous schedule Start Event (see [Asynchronous schedule traversal: Start Event](#) for the definition of the Start Event). The Asynchronous Schedule should have at most one queue head marked as the head (see [Figure 52-15](#)).

The following figure illustrates an example state machine that satisfies the operational requirements of the host controller detecting the first pass through the Asynchronous Schedule. This state machine is maintained internal to the host controller and is only used to gate reloading of the nak counter during the queue head traversal state: *Execute Transaction* (see the figure below). The host controller does not perform the nak counter reload operation if the *RL* field (see [Table 52-25](#)) is set to zero.



**Figure 52-17. Example HC State Machine for Controlling Nak Counter Reloads**

#### 52.4.3.9.1.1 Wait for List Head

This is the initial state.

The state machine enters this state from Wait for Start Event when a start event as defined in [Asynchronous schedule traversal: Start Event](#) occurs.

The purpose of this state is to wait for the first observation of the head of the Asynchronous Schedule.

This occurs when the host controller fetches a queue head whose *H-bit* is set to one.

#### 52.4.3.9.1.2 Do Reload

This state is entered from the Wait for List Head state when the host controller fetches a queue head with the *H-bit* set to one. While in this state, the host controller performs nak counter reloads for every queue head visited that has a non-zero nak reload value (*RL*) field.

### 52.4.3.9.1.3 Wait for Start Event

This state is entered from the *Do Reload* state when a queue head with the *H-bit* set to one is fetched. While in this state, the host controller does not perform nak counter reloads.

## 52.4.3.10 Managing Control/Bulk/Interrupt Transfers through Queue Heads

This section presents an overview of how the host controller interacts with queuing data structures.

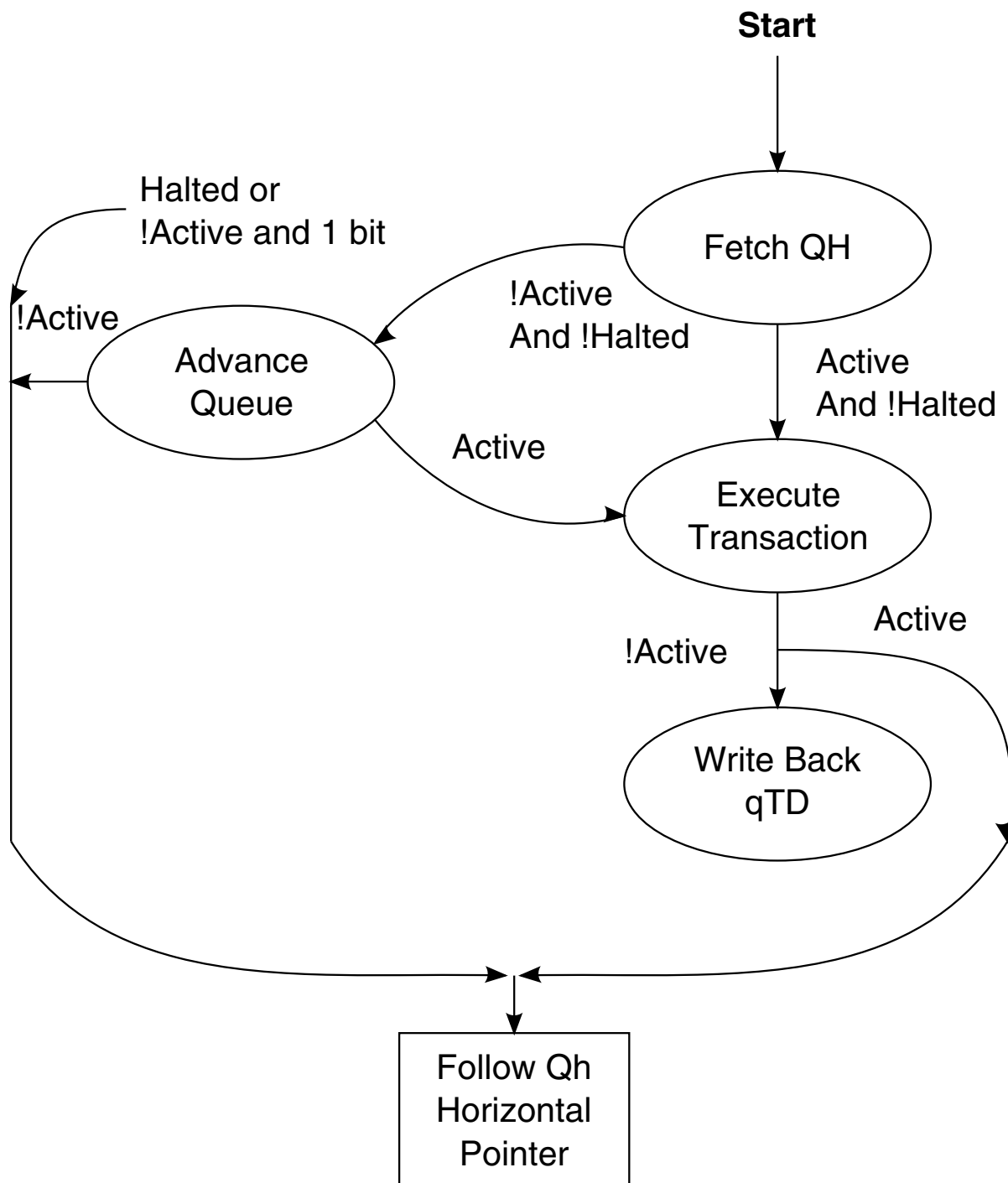
Queue heads use the Queue Element Transfer Descriptor (qTD) structure. One queue head is used to manage the data stream for one endpoint. The queue head structure contains static endpoint characteristics and capabilities. It also contains a working area from where individual bus transactions for an endpoint are executed (see Overlay area defined in [Table 52-25](#)). Each qTD represents one or more bus transactions, which is defined in the context of this specification as a *transfer*.

The general processing model for the host controller's use of a queue head is simple:

- read a queue head,
- execute a transaction from the overlay area,
- write back the results of the transaction to the overlay area,
- move to the next queue head.

If the host controller encounters errors during a transaction, the host controller sets one (or more) of the error reporting bits in the queue head's *Status* field. The *Status* field accumulates all errors encountered during the execution of a qTD (for example, the error bits in the queue head *Status* field are 'sticky' until the transfer (qTD) has completed). This state is always written back to the source qTD when the transfer is complete. On transfer (for example, buffer or halt conditions) boundaries, the host controller must auto-advance (without software intervention) to the next qTD. Additionally, the hardware must be able to halt the queue so no additional bus transactions occurs for the endpoint and the host controller does not advance the queue.

An example host controller operational state machine of a queue head traversal is illustrated in the following figure. This state machine is a model for how a host controller should traverse a queue head. The host controller must be able to advance the queue from the *Fetch QH* state in order to avoid all hardware/software race conditions. This simple mechanism allows software to simply link qTDs to the queue head and *activate* them, then the host controller always *find* them if/when they are reachable. The figure below illustrates the Host Controller Queue Head Traversal State Machine.



**Figure 52-18. Host Controller Queue Head Traversal State Machine**

This traversal state machine applies to all queue heads, regardless of transfer type or whether split transactions are required. The following sections describe each state. Each state description describes the entry criteria. The Execute Transaction state (see [Execute](#)

[Transaction](#) ) describes the basic requirements for all endpoints. [Split Transactions for Asynchronous Transfers](#) and [Split Transaction Interrupt](#) describe details of the required extensions to the Execute Transaction state for endpoints requiring split transactions.

### NOTE

Prior to software placing a queue head into either the periodic or asynchronous list, software must ensure the queue head is properly initialized. Minimally, the queue head should be initialized to the following (see Section Queue Head for layout of a queue head):

Valid static endpoint state.

- For the very first use of a queue head, software may zero-out the queue head transfer overlay, then set the *Next qTD Pointer* field value to reference a valid qTD.

#### 52.4.3.10.1 Fetch Queue Head

A queue head can be referenced from the physical address stored in the ASYNCLISTADDR Register (see [Next Asynch. Address \(USBC\\_n\\_ASYNCLISTADDR\)](#))/[Endpoint List Address \(USBC\\_n\\_ENDPTLISTADDR\)](#)). Additionally, it may be referenced from the *Next LinkPointer* field of an iTD, siTD, FSTN or another Queue Head. If the referencing link pointer has the *Typ* field set to indicate a queue head, it is assumed to reference a queue head structure as defined in [Table 52-25](#).

While in this state, the host controller performs operations to implement empty schedule detection (see [Empty Asynchronous Schedule Detection](#) ) and Nak Counter reloads (see [Operational Model for Nak Counter](#)). After the queue head has been fetched, the host controller conducts the following queries for empty schedule detection:

- If queue head is not an interrupt queue head (that is *S-mask* is zero), and
- The *H-bit* is one, and
- The *Reclamation* bit in the USBSTS register is zero.

When these criteria are met, the host controller stops traversing the asynchronous list (as described in [Empty Asynchronous Schedule Detection](#) ). When the criteria are not met, the host controller continues schedule traversal. If the queue head is not an interrupt and the *H-bit* is one and the *Reclamation* bit is one, then the host controller sets the *Reclamation* bit in the USBSTS register to zero before completing this state. The operations for reloading of the Nak Counter are described in detail in [Operational Model for Nak Counter](#).

This state is complete when the queue head has been read on-chip.



### 52.4.3.10.2 Advance Queue

To advance the queue, the host controller must find the next qTD, adjust pointers, perform the overlay and write back the results to the queue head.

This state is entered from the FetchQHD state if the overlay *Active* and *Halt* bits are set to zero. On entry to this state, the host controller determines which next pointer to use to fetch a qTD, fetches a qTD and determines whether or not to perform an overlay.

#### NOTE

If the *I-bit* is one and the *Active* bit is zero, the host controller immediately skips processing of this queue head, exits this state and uses the horizontal pointer to the next schedule data structure. If the field *Bytes to Transfer* is not zero and the *T-bit* in the *Alternate Next qTD Pointer* is set to zero, then the host controller uses the *Alternate Next qTD Pointer*. Otherwise, the host controller uses the *NextqTD Pointer*. If *NextqTD Pointer's T-bit* is set to one, then the host controller exits this state and uses the horizontal pointer to the next schedule data structure.

Using the selected pointer the host controller fetches the referenced qTD. If the fetched qTD has its *Active* bit set to one, the host controller moves the pointer value used to reach the qTD (*Next* or *Alternate Next*) to the *Current qTD Pointer* field, then performs the overlay. If the fetched qTD has its *Active* bit set to zero, the host controller aborts the queue advance and follows the queue head's horizontal pointer to the next schedule data structure.

The host controller performs the overlay based on the following rules:

- The value of the data toggle (*dt*) field in the overlay area depends on the value of the *data toggle control (dtc)* bit (see [Table 52-27](#)).
- If the *EPS* field indicates the endpoint is a high-speed endpoint, the *Ping* state field is preserved by the host controller. The value of this field is not changed as a result of the overlay.
- *C-prog-mask* field is set to zero (field from incoming qTD is ignored, as is the current contents of the overlay area).
- *Frame Tag* field is set to zero (field from incoming qTD is ignored, as is the current contents of the overlay area).
- *NakCnt* field in the overlay area is loaded from the *RL* field in the queue head's Static Endpoint State.
- All other areas of the overlay are set by the incoming qTD.

The host controller exits this state when it has committed the write to the queue head.

### 52.4.3.10.3 Execute Transaction

The host controller enters this state from the Fetch Queue Head state only if the *Active* bit in *Status* field of the queue head is set to one.

On entry to this state, the host controller executes a few pre-operations, then checks some pre-condition criteria before committing to executing a transaction for the queue head.

The pre-operations performed and the pre-condition criteria depend on whether the queue head is an interrupt endpoint. The host controller can determine that a queue head is an interrupt queue head when the queue head's *S-mask* field contains a non-zero value. It is the responsibility of software to ensure the *S-mask* field is appropriately initialized based on the transfer type. There are other criteria that must be met if the *EPS* field indicates that the endpoint is a low- or full-speed endpoint, see [Split Transactions for Asynchronous Transfers](#) and [Split Transaction Interrupt](#).

#### 52.4.3.10.3.1 Interrupt Transfer Pre-condition Criteria

If the queue head is for an interrupt endpoint (for example, non-zero *S-mask* field), then the FRINDEX[2:0] field must identify a bit in the *S-mask* field that has one in it.

For example, an *S-mask* value of 00100000b would evaluate to true only when FRINDEX[2:0] is equal to 101b. If this condition is met then the host controller considers this queue head for a transaction.

#### 52.4.3.10.3.2 Asynchronous Transfer Pre-operations and Pre-condition Criteria

If the queue head is not for an interrupt endpoint (for example, zero *S-mask* field), then the host controller performs one pre-operation and then evaluates one pre-condition criteria.

The pre-operation is:

Checks the Nak counter reload state ([Operational Model for Nak Counter](#)). It may be necessary for the host controller to reload the Nak Counter field. The reload is performed at this time.

The pre-condition evaluated is:

- Whether or not the *NakCnt* field has been reloaded, the host controller checks the value of the *NakCnt* field in the queue head. If *NakCnt* is non-zero, or if the *Reload Nak Counter* field is zero, then the host controller considers this queue head for a transaction.

### 52.4.3.10.3.3 Transfer Type Independent Pre-operations

Regardless of the transfer type, the host controller always performs at least one pre-operation and evaluates one pre-condition. The pre-operation is:

- A host controller internal transaction (down) counter *qHTransactionCounter* is loaded from the queue head's *Mult* field. A host controller implementation is allowed to ignore this for queue heads on the asynchronous list. It is mandatory for interrupt queue heads. Software should ensure that the *Mult* field is set appropriately for the transfer type.

The pre-conditions evaluated are:

- The host controller determines whether there is enough time in the micro-frame to complete this transaction (see [Transaction Fit - A Best-Fit Approximation Algorithm](#) for an example evaluation method). If there is not enough time to complete the transaction, the host controller exits this state.
- If the value of *qHTransactionCounter* for an interrupt endpoint is zero, then the host controller exits this state.

When the pre-operations are complete and pre-conditions are met, the host controller sets the *Reclamation* bit in the USBSTS register to one and then begins executing one or more transactions using the endpoint information in the queue head. The host controller iterates *qHTransactionCounter* times in this state executing transactions. After each transaction is executed, *qHTransactionCounter* is decremented by one. The host controller exits this state when one of the following events occurs:

- The *qHTransactionCounter* decrements to zero, or
- The endpoint responds to the transaction with any handshake other than an ACK,<sup>4</sup> or
- The transaction experiences a transaction error, or
- The *Active* bit in the queue head goes to zero, or
- There is not enough time in the micro-frame left to execute the next transaction (see [Transaction Fit - A Best-Fit Approximation Algorithm](#) ) for example method for implementing the frame boundary test).

#### NOTE

For a high-bandwidth interrupt OUT endpoint, the host controller may optionally immediately retry the transaction if it fails.

The results of each transaction is recorded in the on-chip overlay area. If data was successfully moved during the transaction, the transfer state in the overlay area is advanced. To advance queue head's transfer state, the *Total Bytes to Transfer* field is decremented by the number of bytes moved in the transaction, the data toggle bit (*dt*) is toggled, the current page offset is advanced to the next appropriate value (for example,

advanced by the number of bytes successfully moved), and the *C\_Page* field is updated to the appropriate value (if necessary). See [Buffer Pointer List Use for Data Streaming with qTDs](#) .

### NOTE

The *Total Bytes To Transfer* field may be zero when all the other criteria for executing a transaction are met. When this occurs, the host controller executes zero-length transaction to the endpoint. If the *PID\_Code* field indicates an IN transaction and the device delivers data, the host controller detects a packet babble condition, set the *babble* and *halted* bits in the *Status* field, set the *Active* bit to zero, write back the results to the source qTD, then exit this state.

In the event an IN token receives a data PID mismatch response, the host controller must ignore the received data (for example not advance the transfer state for the bytes received). Additionally, if the endpoint is an interrupt IN, then the host controller must record that the transaction occurred (for example, decrement *qHTransactionCounter*). It is recommended (but not required) the host controller continue executing transactions for this endpoint if the resultant value of *qHTransactionCounter* is greater than one.

If the response to the IN bus transaction is a Nak (or Nyet) and *RL* is non-zero, *NakCnt* is decremented by one. If *RL* is zero, then no write-back by the host controller is required (for a transaction receiving a Nak or Nyet response and the value of *CErr* did not change). Software should set the *RL* field to zero if the queue head is an interrupt endpoint. Host controller hardware is not required to enforce this rule or operation.

After the transaction has finished and the host controller has completed the post processing of the results (advancing the transfer state and possibly *NakCnt*, the host controller writes back the results of the transaction to the queue head's overlay area in main memory).

The number of bytes moved during an IN transaction depends on how much data the device endpoint delivers. The maximum number of bytes a device can send is *MaximumPacket Size*. The number of bytes moved during an OUT transaction is either *Maximum Packet Length* bytes or *Total Bytes to Transfer*, whichever is less.

If there was a transaction error during the transaction, the transfer state (as defined above) is not advanced by the host controller. The *CErr* field is decremented by one and the status field is updated to reflect the type of error observed. Transaction errors are summarized in [Transaction Error](#) .

The following events causes the host controller to clear the *Active* bit in the queue head's overlay status field. When the *Active* bit transitions from one to zero, the transfer in the overlay is considered complete. The reason for the transfer completion (clearing the *Active* bit) determines the next state.

- *CErr* field decrements to zero. When this occurs the *Halted* bit is set to one and *Active* is set to zero. This results in the hardware not advancing the queue and the pipe halts. Software must intercede to recover.
- The device responds to the transaction with a STALL PID. When this occurs, the *Halted* bit is set to one and the *Active* bit is set to zero. This results in the hardware not advancing the queue and the pipe halts. Software must intercede to recover.
- The *Total Bytes to Transfer* field is zero after the transaction completes.
  - For a zero length transaction, it was zero before the transaction was started. When this condition occurs, the *Active* bit is set to zero.
- The PID code is an IN, and the number of bytes moved during the transaction is less than the *Maximum Packet Length*. When this occurs, the *Active* bit is set to zero and a short packet condition exists. The short-packet condition is detected during the Advance Queue state. Refer to [Split Transactions](#) for additional rules for managing low- and full-speed transactions.

With the exception of a NAK response (when *RL* field is zero), the host controller always writes the results of the transaction back to the overlay area in main memory. This includes when the transfer completes. For a high-speed endpoint, the queue head information written back includes minimally the following fields: The *PID Code* field indicates an IN and the device sends more than the expected number of bytes (for example *Maximum Packet Length* or *Total Bytes to Transfer* bytes, whichever is less) (for example a packet babble). This results in the host controller setting the *Halted* bit to one.

- NakCnt, dt, Total Bytes to Transfer, C\_Page, Status, CERR, and Current Offset

For a low- or full-speed device the queue head information written back also includes the fields:

- C-prog-mask, FrameTag and S-bytes.

The duration of this state depends on the time it takes to complete the transaction(s) and the status write to the overlay is committed.

#### 52.4.3.10.3.4 Halting a Queue Head

A halted endpoint is defined only for the transfer types that are managed through queue heads (control, bulk and interrupt).

The following events indicate that the endpoint has reached a condition where no more activity can occur without intervention from the driver:

- An endpoint may return a STALL handshake during a transaction,
- A transaction had three consecutive error conditions, or
- A Packet Babble error occurs on the endpoint.

When any of these events occur (for a queue head) the Host Controller halts the queue head and set the USBERRINT status bit in the USB\_n\_USBSTS register to one. To halt the queue head, the *Active* bit is set to zero and the *Halted* bit is set to one. There may be other error status bits that are set when a queue is halted. The host controller always writes back the overlay area to the source qTD when the transfer is complete, regardless of the reason (normal completion, short packet or halt). The host controller does not advance the transfer state on a transaction that results in a *Halt* condition (for example no updates necessary for *Total Bytes to Transfer*, *C\_Page*, *Current Offset*, and *dt*). The host controller must update *CErr* as appropriate. When a queue head is halted, the *USB Error Interrupt* bit in the USB\_n\_USBSTS register is set to one. If the *USB Error Interrupt Enable* bit in the USB\_n\_USBINTR register is set to one, a hardware interrupt is generated at the next interrupt threshold.

#### 52.4.3.10.3.5 Asynchronous Schedule Park Mode

Asynchronous Schedule Park mode is a special execution mode that can be enabled by system software, where the host controller is permitted to execute more than one bus transaction from a high-speed queue head in the Asynchronous schedule before continuing horizontal traversal of the Asynchronous schedule.

This feature has no effect on queue heads or other data structures in the Periodic schedule. This feature is similar in intent as the *Mult* feature that is used in the Periodic schedule. Where-as the *Mult* feature is a characteristic that is tunable for each endpoint; park-mode is a policy that is applied to all high-speed queue heads in the asynchronous schedule. It is essentially the specification of an iterator for consecutive bus transactions to the same endpoint. All of the rules for managing bus transactions and the results of those as defined in [Execute Transaction](#) apply. This feature merely specifies how many consecutive times the host controller is permitted to execute from the same queue head before moving to the next queue head in the Asynchronous List. This feature should allow the host controller to attain better bus utilization for those devices that are capable of moving data at maximum rate, while at the same time providing a fair service to all endpoints.

A host controller exports its capability to support this feature to system software by setting the *Asynchronous Schedule Park Capability* bit in the USB\_n\_HCCPARAMs register to one. This information keys system software that the *Asynchronous Schedule Park Mode Enable* and *Asynchronous Schedule Park Mode Count* fields in the USB\_n\_USBCMD register are modifiable. System software enables the feature by writing a one to the *Asynchronous Schedule Park Mode Enable* bit.



When park-mode is not enabled (for example *Asynchronous Schedule Park Mode Enable* bit in the USB\_n\_USBCMD register is zero), the host controller must not execute more than one bus transaction per high-speed queue head, per traversal of the asynchronous schedule. When park-mode is enabled, the host controller must not apply the feature to a queue head whose *EPS* field indicates a Low/Full-speed device (for example only one bus transaction is allowed from each Low/Full-speed queue head per traversal of the asynchronous schedule). Park-mode may only be applied to queue heads in the Asynchronous schedule whose *EPS* field indicates that it is a high-speed device.

The host controller must apply park mode to queue heads whose *EPS* field indicates a high-speed endpoint. The maximum number of consecutive bus transactions a host controller may execute on a high-speed queue head is determined by the value in the *Asynchronous Schedule Park Mode Count* field in the USB\_n\_USBCMD register. Software must not set *Asynchronous Schedule Park Mode Enable* bit to one and also set *Asynchronous Schedule Park Mode Count* field to zero. The resulting behavior is not defined. An example behavioral example describes the operational requirements for the host controller implementing park-mode. This feature does not affect how the host controller handles the bus transaction as defined in [Execute Transaction](#). It only effects how many consecutive bus transactions for the current queue head can be executed. All boundary conditions, error detection and reporting applies as usual. This feature is similar in concept to the use of the *Mult* field for high-bandwidth Interrupt for queue heads in the Periodic Schedule.

The host controller effectively loads an internal down-counter *PM-Count* from *Asynchronous Schedule Park Mode Count* when *Asynchronous Schedule Park Mode Enable* bit is one, and a high-speed queue head is first fetched and meets all the criteria for executing a bus transaction. After the bus transaction, *PM-Count* is decremented. The host controller may continue to execute bus transactions from the current queue head until *PM-Count* goes to zero, an error is detected, the buffer for the current transfer is exhausted or the endpoint responds with a flow-control or STALL handshake.

The following table summarizes the responses that effect whether the host controller continues with another bus transaction for the current queue head.

**Table 52-43. Actions for Park Mode, based on Endpoint Response and Residual Transfer State**

PID	Endpoint Response	Transfer State after Transaction		Action
		PM-Count	Bytes to Transfer	

*Table continues on the next page...*

**Table 52-43. Actions for Park Mode, based on Endpoint Response and Residual Transfer State (continued)**

IN	DATA[0,1] w/Maximum Packet sized data	Not zero	Not Zero	Allowed to perform another bus transaction. <sup>12</sup>
		Not zero	Zero	Retire qTD and move to next QH
		Zero	Don't care	Move to next QH.
	DATA[0,1] w/short packet	Don't care	Don't care	Retire qTD and move to next QH.
	NAK	Don't care	Don't care	Move to next QH.
	STALL, XactErr	Don't care	Don't care	Move to next QH.
OUT	ACK	Not zero	Not Zero	Allowed to perform another bus transaction. <sup>2</sup>
		Not zero	Zero	Retire qTD and move to next QH
		Zero	Don't care	Move to next QH.
	NYET, NAK	Don't care	Don't care	Move to next QH.
	STALL, XactErr	Don't care	Don't care	Move to next QH
PING	ACK	Not Zero	Not Zero	Allowed to perform another bus transaction. <sup>2</sup>
	NAK	Don't care	Don't care	Move to next QH
	STALL, XactErr	Don't care	Don't care	Move to next QH

1. The host controller may continue to execute bus transactions from the current high-speed queue head (if *PM-Count* is not equal to zero), if a PID mismatch is detected (for example expected DATA1 and received DATA0, or visa-versa).
2. This specification does not *require* that the host controller execute another bus transaction when *PM-Count* is non-zero. Implementations are encouraged to make appropriate complexity and performance trade-offs.

#### 52.4.3.10.4 Write Back qTD

This state is entered from the Execute Transaction state when the *Active* bit is set to zero.

The source data for the write-back is the transfer results area of the queue head overlay area (see [Table 52-43](#)).

The host controller uses the *Current qTD Pointer* field as the target address for the qTD.

The queue head transfer result area is written back to the transfer result area of the target qTD. This state is also referred to as: qTD retirement. The fields that must be written back to the source qTD include *Total Bytes to Transfer*, *Cerr*, and *Status*.

The duration of this state depends on when the qTD write-back is committed.

#### 52.4.3.10.5 Follow Queue Head Horizontal Pointer

The host controller must use the horizontal pointer in the queue head to the next schedule data structure when any of the following conditions exist:

- If the *Active* bit is one on exit from the Execute Transaction state, or
- When the host controller exits the Write Back qTD state, or



- If the Advance Queue state fails to advance the queue because the target qTD is not active, or
- If the *Halted* bit is one on exit from the Fetch QH state.

There is no functional requirement that the host controller wait until the current transaction is complete before using the horizontal pointer to read the next linked data structure. However, it must wait until the current transaction is complete before executing the next data structure.

#### 52.4.3.10.6 Buffer Pointer List Use for Data Streaming with qTDs

A qTD has an array of buffer pointers, which is used to reference the data buffer for a transfer. This specification requires that the buffer associated with the transfer be *virtually contiguous*.

This means: if the buffer spans more than one physical page, it must obey the following rules (the figure below illustrates an example):

- The first portion of the buffer must begin at some offset in a page and extend through the end of the page.
- The remaining buffer cannot be allocated in small chunks scattered around memory. For each 4 K chunk beyond the first page, each buffer portion matches to a full 4 K page. The final portion, which may only be large enough to occupy a portion of a page, must start at the top of the page and be contiguous within that page.

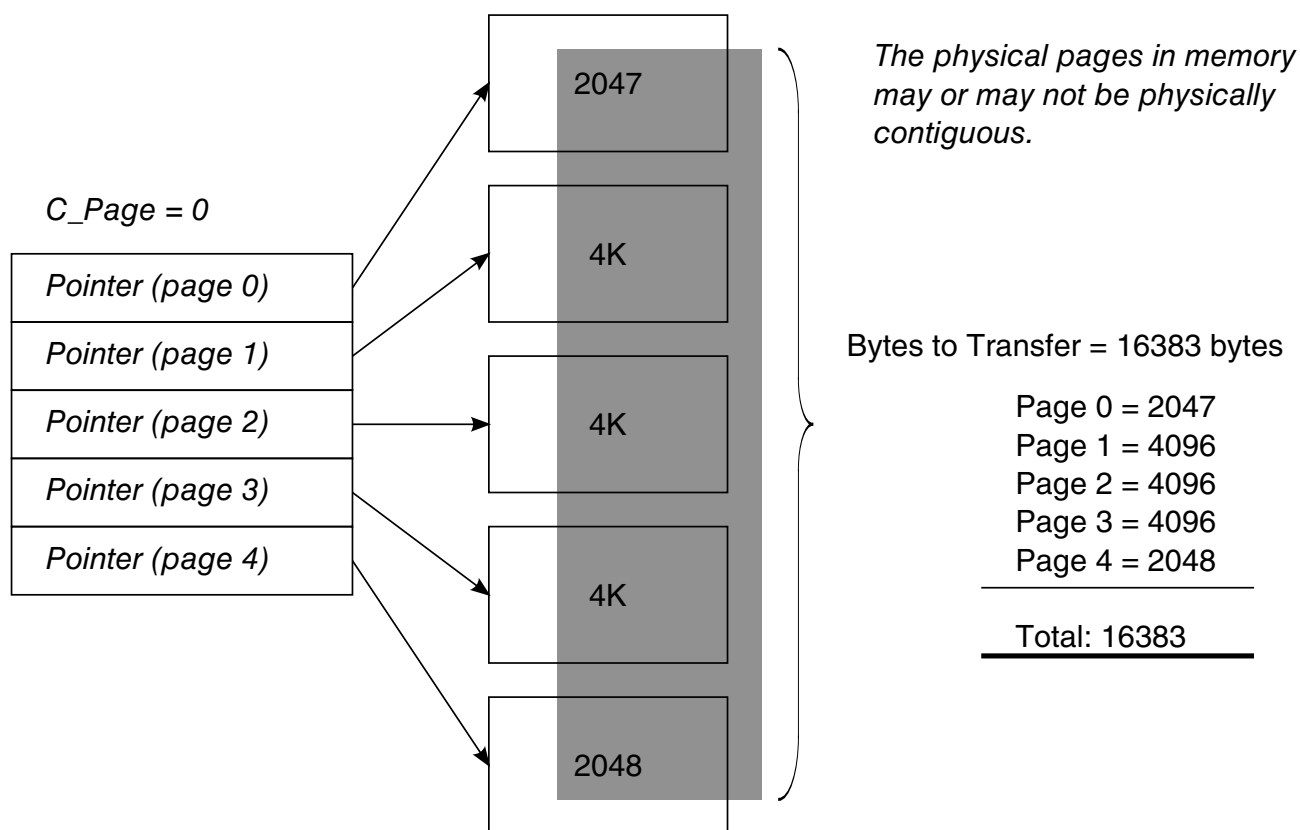
The buffer pointer list in the qTD is long enough to support a maximum transfer size of 20 K bytes. This case occurs when all five buffer pointers are used and the first offset is zero. A qTD handles a 16 Kbyte buffer with any starting buffer alignment.

The host controller uses the field *C\_Page* field as an index value to determine which buffer pointer in the list should be used to start the current transaction. The host controller uses a different buffer pointer for each physical page of the buffer. This is always true, even if the buffer is physically contiguous.

The host controller must detect when the current transaction spans a page boundary and automatically move to the next available buffer pointer in the page pointer list. The next available pointer is reached by incrementing *C\_Page* and pulling the next page pointer from the list. Software must ensure there are sufficient buffer pointers to move the amount of data specified in the *Bytes to Transfer* field.

The following figure illustrates a nominal example of how System software would initialize the buffer pointers list and the *C\_Page* field for a transfer size of 16383 bytes. *C\_Page* is set to zero. The upper 20-bits of Page 0 references the start of the physical

page. *Current Offset* (the lower 12-bits of queue head Dword 7) holds the offset in the page for example 2049 (for example 4096-2047). The remaining page pointers are set to reference the beginning of each subsequent 4 K page.



**Figure 52-19. Example Mapping of qTD Buffer Pointers to Buffer Pages**

For the first transaction on the qTD (assuming a 512-byte transaction), the host controller uses the first buffer pointer (page 0 because *C\_Page* is set to zero) and concatenates the *Current Offset* field. The 512 bytes are moved during the transaction, the *Current Offset* and *Total Bytes to Transfer* are adjusted by 512 and written back to the queue head working area.

During the 4<sup>th</sup> transaction, the host controller needs 511 bytes in page 0 and one byte in page 1. The host controller increments *C\_Page* (to 1) and use the page 1 pointer to move the final byte of the transaction. After the 4<sup>th</sup> transaction, the active page pointer is the page 1 pointer and *Current Offset* has rolled to one, and both are written back to the overlay area. The transactions continue for the rest of the buffer, with the host controller automatically moving to the next page pointer (that is *C\_Page*) when necessary. The three conditions for how the host controller handles *C\_Page*:

- The current transaction does not span a page boundary. The value of *C\_Page* is not adjusted by the host controller.

- The current transaction does span a page boundary. The host controller must detect the page cross condition and advance to the next buffer while streaming data to/from the USB.
- The current transaction completes on a page boundary (that is the last byte moved for the current transaction is the last byte in the page for the current page pointer). The host controller must increment *C\_Page* before writing back status for the transaction.

### NOTE

The only valid adjustment the host controller may make to *C\_Page* is to increment by one.

#### 52.4.3.10.7 Adding Interrupt Queue Heads to the Periodic Schedule

The link path(s) from the periodic frame list to a queue head establishes in which frames a transaction can be executed for the queue head. Queue heads are linked into the periodic schedule so they are polled at the appropriate rate.

System software sets a bit in a queue head's *S-Mask* to indicate which micro-frame within 1 msec period a transaction should be executed for the queue head. Software must ensure that all queue heads in the periodic schedule have *S-Mask* set to a non-zero value. An *S-mask* with zero value in the context of the periodic schedule yields undefined results.

If the desired poll rate is greater than one frame, system software can use a combination of queue head linking and *S-Mask* values to spread interrupts of equal poll rates through the schedule so that the periodic bandwidth is allocated and managed in the most efficient manner possible. Some examples are illustrated in the following table.

**Table 52-44. Example Periodic Reference Patterns for Interrupt Transfers with 2ms Poll Rate**

Frame # Reference Sequence	Description
0, 2, 4, 6, 8, and so on <i>S-Mask</i> = 01h	A queue head for the <i>bInterval</i> of 2 msec (16 micro-frames) is linked into the periodic schedule so that it is reachable from the periodic frame list locations indicated in the previous column. In addition, the <i>S-Mask</i> field in the queue head is set to 01h, indicating that the transaction for the endpoint should be executed on the bus during micro-frame 0 of the frame.
0, 2, 4, 6, 8, and so on <i>S-Mask</i> = 02h	Another example of a queue head with a <i>bInterval</i> of 2 msec is linked into the periodic frame list at exactly the same interval as the previous example. However, the <i>S-Mask</i> is set to 02h indicating that the transaction for the endpoint should be executed on the bus during micro-frame 1 of the frame.

### 52.4.3.10.8 Managing Transfer Complete Interrupts from Queue Heads

The host controller sets an interrupt to be signaled at the next interrupt threshold when the completed transfer (qTD) has an *Interrupt on Complete (IOC)* bit set to one, or whenever a transfer (qTD) completes with a short packet.

If system software needs multiple qTDs to complete a client request (that is like a control transfer) the intermediate qTDs do not require interrupts. System software may only need a single interrupt to notify it that the complete buffer has been transferred. System software may set IOC's to occur more frequently. A motivation for this may be that it wants early notification so that interface data structures can be re-used in a timely manner.

### 52.4.3.11 Ping Control

USB 2.0 defines an addition to the protocol for high-speed devices called Ping. Ping is required for all USB 2.0 High-speed bulk and control endpoints.

Ping is not allowed for a split-transaction stream. This extension to the protocol eliminates the bad side-effects of Naking OUT endpoints. The *Status* field has a *Ping State* bit, which the host controller uses to determine the *next* actual PID it uses in the next transaction to the endpoint (see the table below).

The Ping State bit is only managed by the host controller for queue heads that meet the following criteria:

- Queue head is not an interrupt and
- *EPS* field equals High-Speed and
- *PIDCode* field equals OUT

The following table illustrates the state transition table for the host controller's responsibility for maintaining the PING protocol. Refer to Chapter 8 in the USB Specification Revision 2.0 for detailed description on the Ping protocol.

**Table 52-45. Ping Control State Transition Table**

Event			
Current	Host	Device	Next
Do Ping	PING	Nak	Do Ping
Do Ping	PING	Ack	Do OUT
Do Ping	PING	XactErr <sup>1</sup>	Do Ping
Do Ping	PING	Stall	N/C <sup>2</sup> Do
OUT	OUT	Nak	Do Ping
Do OUT	OUT	Nyet	Do Ping

*Table continues on the next page...*

**Table 52-45. Ping Control State Transition Table (continued)**

Do OUT	OUT	Ack	Do OUT
Do OUT	OUT	XactErr <sup>1</sup>	Do Ping
Do OUT	OUT	Stall	N/C <sup>2</sup>

1. Transaction Error (XactErr) is any time the host misses the handshake.
2. No transition change required for the Ping State bit. The Stall handshake results in the endpoint being halted (for example Active set to zero and Halt set to one). Software intervention is required to restart queue. 3 A Nyet response to an OUT means that the device has accepted the data, but cannot receive any more at this time. Host must advance the transfer state and additionally, transition the Ping State bit to Do Ping. The Ping State bit has the following encoding:

**Table 52-46. Ping State bit Encoding**

Value	Meaning
0B	Do OUT The host controller uses an OUT PID during the next bus transaction to this endpoint.
1B	Do Ping The host controller uses a PING PID during the next bus transaction to this endpoint.

The defined ping protocol (see USB 2.0 Specification, Chapter 8) allows the host to be *imprecise* on the initialization of the ping protocol (that is start in *Do OUT* when we don't know whether there is space on the device or not). The host controller manages the *Ping State* bit. System software sets the initial value in the queue head when it initializes a queue head. The host controller preserves the *Ping State* bit across all queue advancements. This means that when a new qTD is written into the queue head overlay area, the previous value of the *Ping State* bit is preserved.

### 52.4.3.12 Split Transactions

USB 2.0 defines extensions to the bus protocol for managing USB 1.x data streams through USB 2.0 Hubs.

This section describes how the host controller uses the interface data structures to manage data streams with full- and low-speed devices, connected below USB 2.0 hub, utilizing the split transaction protocol.

Refer to USB 2.0 Specification for the complete definition of the split transaction protocol. Full- and Low-speed devices are enumerated identically as high-speed devices, but the transactions to the Full- and Low-speed endpoints use the split-transaction protocol on the high-speed bus. The split transaction protocol is an encapsulation of (or wrapper around) the Full- or Low-speed transaction. The high-speed wrapper portion of the protocol is addressed to the USB 2.0 Hub and Transaction Translator below which the Full- or Low-speed device is attached.

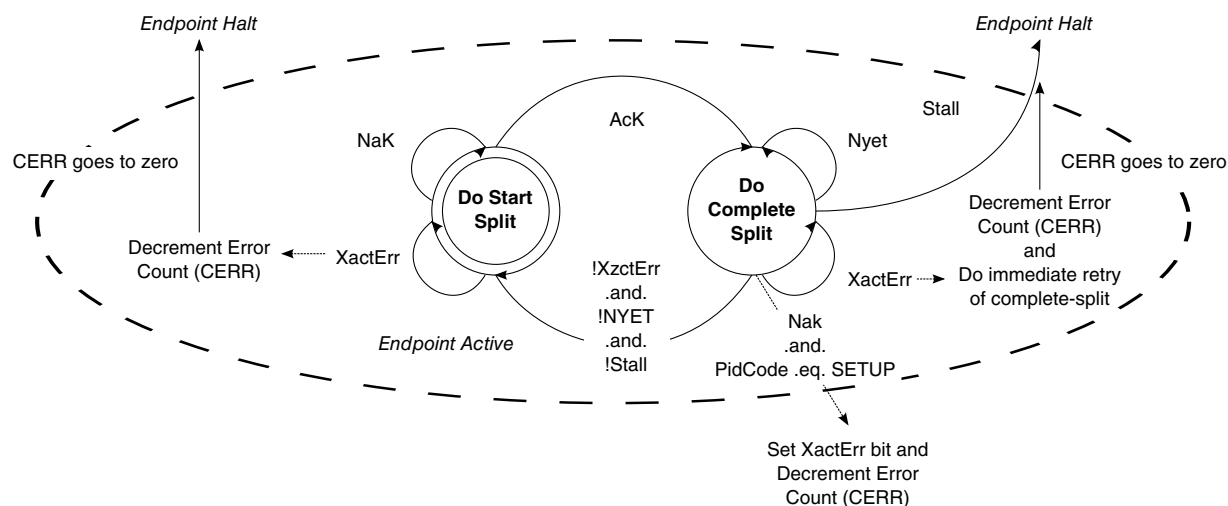
The EHCI interface uses dedicated data structures for managing full-speed isochronous data streams (see [Split Transaction Isochronous Transfer Descriptor \(siTD\)](#)). Control, Bulk and Interrupt are managed using the queuing data structures (see [Queue Head](#)). The interface data structures need to be programmed with the device address and the Transaction Translator number of the USB 2.0 Hub operating as the Low-/Full-speed host controller for this link. The following sections describe the details of how the host controller must process and manage the split transaction protocol.

### 52.4.3.12.1 Split Transactions for Asynchronous Transfers

A queue head in the asynchronous schedule with an *EPS* field indicating a full-or low-speed device indicates to the host controller that it must use split transactions to stream data for this queue head.

All full-speed bulk and full-, low-speed control are managed through queue heads in the asynchronous schedule.

Software must initialize the queue head with the appropriate device address and port number for the transaction translator that is serving as the full/low-speed host controller for the links connecting the endpoint. Software must also initialize the split transaction state bit (*SplitXState*) to Do-Start-Split. Finally, if the endpoint is a control endpoint, then system software must set the *Control Transfer Type (C)* bit in the queue head to one. If this is not a control transfer type endpoint, the *C* bit must be initialized by software to be zero. This information is used by the host controller to properly set the Endpoint Type (ET) field in the split transaction bus token. When the *C* bit is zero, the split transaction token's ET field is set to indicate a bulk endpoint. When the *C* bit is one, the split transaction token's ET field is set to indicate a control endpoint. Refer to Chapter 8 of USB Specification Revision 2.0 for details.



**Figure 52-20. Host Controller Asynchronous Schedule Split-Transaction State Machine**

### 52.4.3.12.1.1 Asynchronous - Do Start Split

This is the state which software must initialize a full- or low-speed asynchronous queue head. This state is entered from the Do Complete Split state only after a complete-split transaction receives a valid response from the transaction translator that is not a Nyet handshake.

For queue heads in this state, the host controller executes a start-split transaction to the appropriate transaction translator. If the bus transaction completes without an error and *PidCode* indicates an IN or OUT transaction, then the host controller reloads the error counter (*CErr*). If it is a successful bus transaction and the *PidCode* indicates a SETUP, the host controller does not reload the error counter. If the transaction translator responds with a Nak, the queue head is left in this state, and the host controller proceeds to the next queue head in the asynchronous schedule.

If the host controller times out the transaction (no response, or bad response) the host controller decrements *Cerr* and proceeds to the next queue head in the asynchronous schedule.

### 52.4.3.12.1.2 Asynchronous - Do Complete Split

This state is entered from the Do Start Split state only after a start-split transaction receives an Ack handshake from the transaction translator.

For queue heads in this state, the host controller executes a complete-split transaction to the appropriate transaction translator. If the transaction translator responds with a Nyet handshake, the queue head is left in this state, the error counter is reset and the host controller proceeds to the next queue head in the asynchronous schedule. When a Nyet handshake is received for a bus transaction where the queue head's *PidCode* indicates an IN or OUT, the host controller reloads the error counter (*CErr*). When a Nyet handshake is received for a complete-split bus transaction where the queue head's *PidCode* indicates a SETUP, the host controller must not adjust the value of *CErr*.

Independent of *PIDCode*, the following responses have the effects:

- Transaction Error (XactErr). Timeout or data CRC failure, and so on. The error counter (*Cerr*) is decremented by one and the complete split transaction is *immediately* retried (if possible). If there is not enough time in the micro-frame to execute the retry, the host controller **MUST** ensure that the next time the host controller begins executing from the Asynchronous schedule, it must begin executing from this queue head. If another start-split (for some other endpoint) is sent to the transaction translator before the complete-split is really completed, the transaction translator could dump the results (which were never delivered to the host). This is why the core specification states the retries must be immediate. A method to

accomplish this behavior is to not advance the asynchronous schedule. When the host controller returns to the asynchronous schedule in the next micro-frame, the first transaction from the schedule is the retry for this endpoint.

If *Cerr* went to zero, the host controller must halt the queue.

- **NAK.** The target endpoint Nak'd the full- or low-speed transaction. The state of the transfer is not advanced and the state is exited. If the *PidCode* is a SETUP, then the Nak response is a protocol error. The *XactErr* status bit is set to one and the *CErr* field is decremented.
- **STALL.** The target endpoint responded with a STALL handshake. The host controller sets the *halt* bit in the status byte, retires the qTD but does not attempt to advance the queue.

If the *PidCode* indicates an IN, then any of following responses are expected:

- **DATA0/1.** On reception of data, the host controller ensures the PID matches the expected data toggle and checks CRC. If the packet is *good*, the host controller advances the state of the transfer, for example move the data pointer by the number of bytes received, decrement *BytesToTransfer* field by the number of bytes received, and toggle the *dt* bit. The host controller then exit this state. The response and advancement of transfer may trigger other processing events, such as retirement of the qTD and advancement of the queue.

If the data sequence PID does not match the expected, the data is ignored, the transfer state is not advanced and this state is exited. If the *PidCode* indicates an OUT/SETUP, then any of following responses are expected:

- **ACK.** The target endpoint accepted the data, so the host controller must advance the state of the transfer. The *Current Offset* field is incremented by *Maximum Packet Length* or *Bytes to Transfer*, whichever is less. The field *Bytes To Transfer* is decremented by the same amount and the data toggle bit (*dt*) is toggled. The host controller then exit this state.
- Advancing the transfer state may cause other processing events such as retirement of the qTD and advancement of the queue (see [Managing Control/Bulk/Interrupt Transfers through Queue Heads](#)).

#### 52.4.3.12.2 Split Transaction Interrupt

Split-transaction Interrupt-IN/OUT endpoints are managed through the same data structures used for high-speed interrupt endpoints. They both co-exist in the periodic schedule.



Queue heads/qTDs offer the set of features required for reliable data delivery, which is characteristic to interrupt transfer types. The split-transaction protocol is managed completely within this defined functional transfer framework. For example, for a high-speed endpoint, the host controller visits a queue head, execute a high-speed transaction (if criteria are met) and advance the transfer state (or not) depending on the results of the entire transaction. For low- and full-speed endpoints, the details of the *execution* phase are different (that is takes more than one bus transaction to complete), but the remainder of the operational framework is intact. This means that the transfer advancement, and so on, occurs as defined in [Managing Control/Bulk/Interrupt Transfers through Queue Heads](#), but only occurs on the completion of a split transaction.

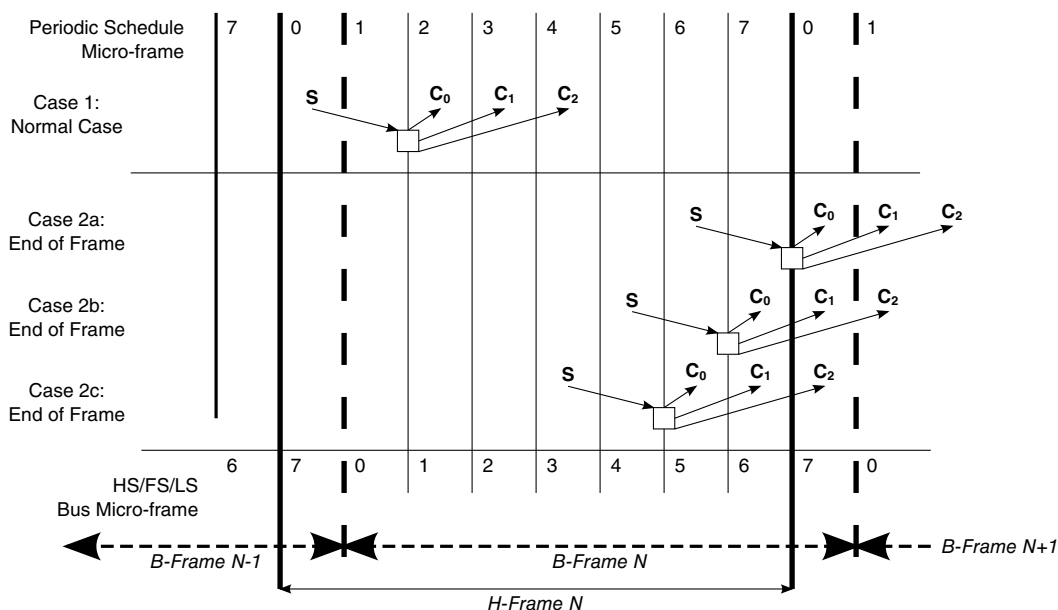
#### 52.4.3.12.2.1 Split Transaction Scheduling Mechanisms for Interrupt

Full- and low-speed Interrupt queue heads have an *EPS* field indicating full- or low-speed and have a non-zero *S-mask* field.

The host controller can detect this combination of parameters and assume the endpoint is a periodic endpoint. Low- and full-speed interrupt queue heads require the use of the split transaction protocol. The host controller sets the Endpoint Type (ET) field in the split token to indicate the transaction is an interrupt. These transactions are managed through a transaction translator's periodic pipeline. Software should not set these fields to indicate the queue head is an interrupt unless the queue head is used in the periodic schedule.

System software manages the per/transaction translator periodic pipeline by budgeting and scheduling exactly during which micro-frames the start-splits and complete-splits for each endpoint occurs. The characteristics of the transaction translator are such that the high-speed transaction protocol must execute during explicit micro-frames, or the data or response information in the pipeline is lost.

The following figure illustrates the general scheduling boundary conditions that are supported by the EHCI periodic schedule and queue head data structure. The S and <sup>C</sup>X labels indicate micro-frames where software can schedule start-splits and complete splits (respectively).

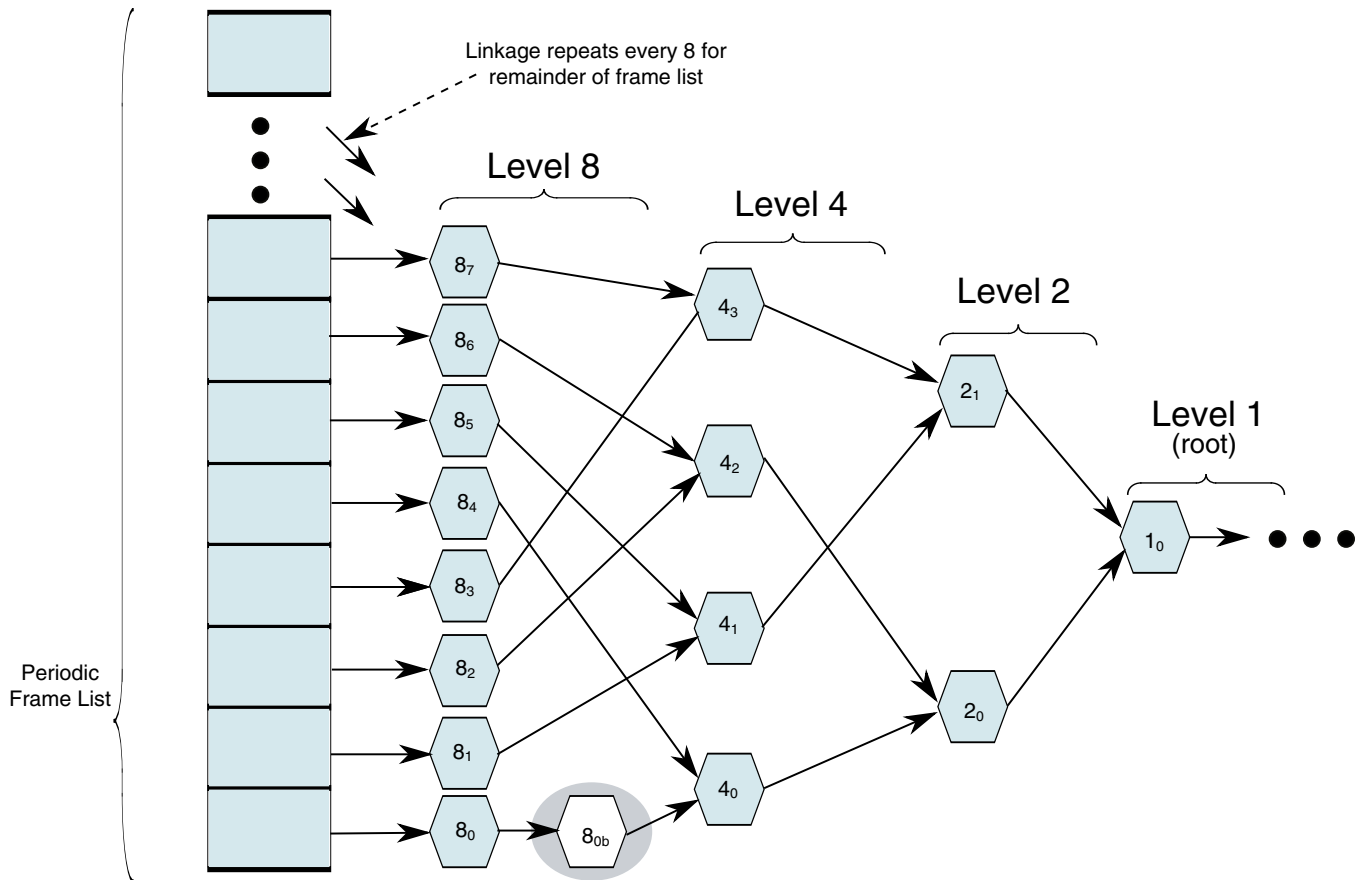


**Figure 52-21. Split Transaction, Interrupt Scheduling Boundary Conditions**

The scheduling cases are:

- Case 1: The normal scheduling case is where the entire split transaction is completely bounded by a frame (*H-Frame* in this case).
- Case 2a through Case 2c: The USB 2.0 Hub pipeline rules states clearly, when and how many complete-splits must be scheduled to account for earliest to latest execution on the full/low-speed link. The complete-splits may span the *H-Frame* boundary when the start-split is in micro-frame 4 or later. When this occurs, the *H-Frame* to *B-Frame* alignment requires that the queue head be reachable from consecutive periodic frame list locations. System software cannot build an efficient schedule that satisfies this requirement unless it uses FSTNs.

The figure below illustrates the general layout of the periodic schedule.



**Figure 52-22. General Structure of EHCI Periodic Schedule Utilizing Interrupt Spreading**

The periodic frame list is effectively the leaf level a binary tree, which is always traversed leaf to root. Each level in the tree corresponds to a  $2^N$  poll rate. Software can efficiently manage periodic bandwidth on the USB by *spreading* interrupt queue heads that have the same poll rate requirement across all the available paths from the frame list. For example, system software can schedule eight poll rate 8 queue heads and account for them once in the high-speed bus bandwidth allocation.

When an endpoint is allocated an execution footprint that spans a frame boundary, the queue head for the endpoint must be reachable from consecutive locations in the frame list. An example would be if 8<sub>0b</sub> where such an endpoint. Without additional support on the interface, to get 8<sub>0b</sub> reachable at the correct time, software would have to link 8<sub>1</sub> to 8<sub>0b</sub>. It would then have to move 4<sub>1</sub> and everything linked after into the same path as 4<sub>0</sub>. This upsets the integrity of the binary tree and disallows the use of the spreading technique.

FSTN data structures are used to preserve the integrity of the binary-tree structure and enable the use of the spreading technique. [Host Controller Operational Model for FSTNs](#) defines the hardware and software operational model requirements for using FSTNs.

The following queue head fields are initialized by system software to instruct the host controller when to execute portions of the split-transaction protocol:

- *SplitXState*. This is single bit residing in the *Status* field of a queue head (see [Table 52-23](#)). This bit is used to track the current state of the split transaction.
- *Frame S-mask*. This is a bit-field where-in system software sets a bit corresponding to the micro-frame (within an *H-Frame*) that the host controller should execute a start-split transaction. This is always qualified by the value of the *SplitXState* bit in the *Status* field of the queue head. For example, referring to [Figure 52-21](#), case one, the *S-mask* would have a value of 00000001b indicating that if the queue head is traversed by the host controller, and the *SplitXState* indicates Do\_Start, and the current micro-frame as indicated by FRINDEX[2:0] is 0, then execute a start-split transaction.
- *Frame C-mask*. This is a bit-field where system software sets one or more bits corresponding to the micro-frames (within an *H-Frame*) that the host controller should execute complete-split transactions. The interpretation of this field is always qualified by the value of the *SplitXState* bit in the *Status* field of the queue head. For example, referring to [Figure 52-21](#), case one, the *C-mask* would have a value of 00011100b indicating that if the queue head is traversed by the host controller, and the *SplitXState* indicates Do\_Complete, and the current micro-frame as indicated by FRINDEX[2:0] is 2, 3, or 4, then execute a complete-split transaction. It is software's responsibility to ensure that the translation between *H-Frames* and *B-Frames* is correctly performed when setting bits in *S-mask* and *C-mask*.

#### 52.4.3.12.2.2 Host Controller Operational Model for FSTNs

The FSTN data structure is used to manage Low/Full-speed interrupt queue heads that need to be reached from consecutive frame list locations (that is boundary cases 2a through 2c).

An FSTN is essentially a *back pointer*, similar in intent to the back pointer field in the siTD data structure (see [siTD Back Link Pointer](#)).

This feature provides software a simple primitive to save a schedule position, redirect the host controller to traverse the necessary queue heads in the previous frame, then restore the original schedule position and complete normal traversal.

The four components to the use of FSTNs:

- FSTN data structure.
- A *Save Place* indicator. This is always an FSTN with its *Back Path Link Pointer.T-bit* set to zero.

- A *Restore* indicator. This is always an FSTN with its *Back Path Link Pointer.T-bit* set to one.
- Host controller FSTN traversal rules.

When the host controller encounters an FSTN during micro-frames 2 through 7 it simply follows the node's *Normal Path Link Pointer* to access the next schedule data structure.

### NOTE

The FSTN's *Normal Path Link Pointer.T-bit* may set to one, which the host controller must interpret as the end of periodic list mark.

When the host controller encounters a *Save-Place* FSTN in micro-frames 0 or 1, it saves the value of the *Normal Path Link Pointer* and set an internal flag indicating that it is executing in *Recovery Path* mode. *Recovery Path* mode modifies the host controller's rules for how it traverses the schedule and limits which data structures is considered for execution of bus transactions. The host controller continues executing in *Recovery Path* mode until it encounters a *Restore* FSTN or it determines that it has reached the end of the micro-frame (see details in the list below).

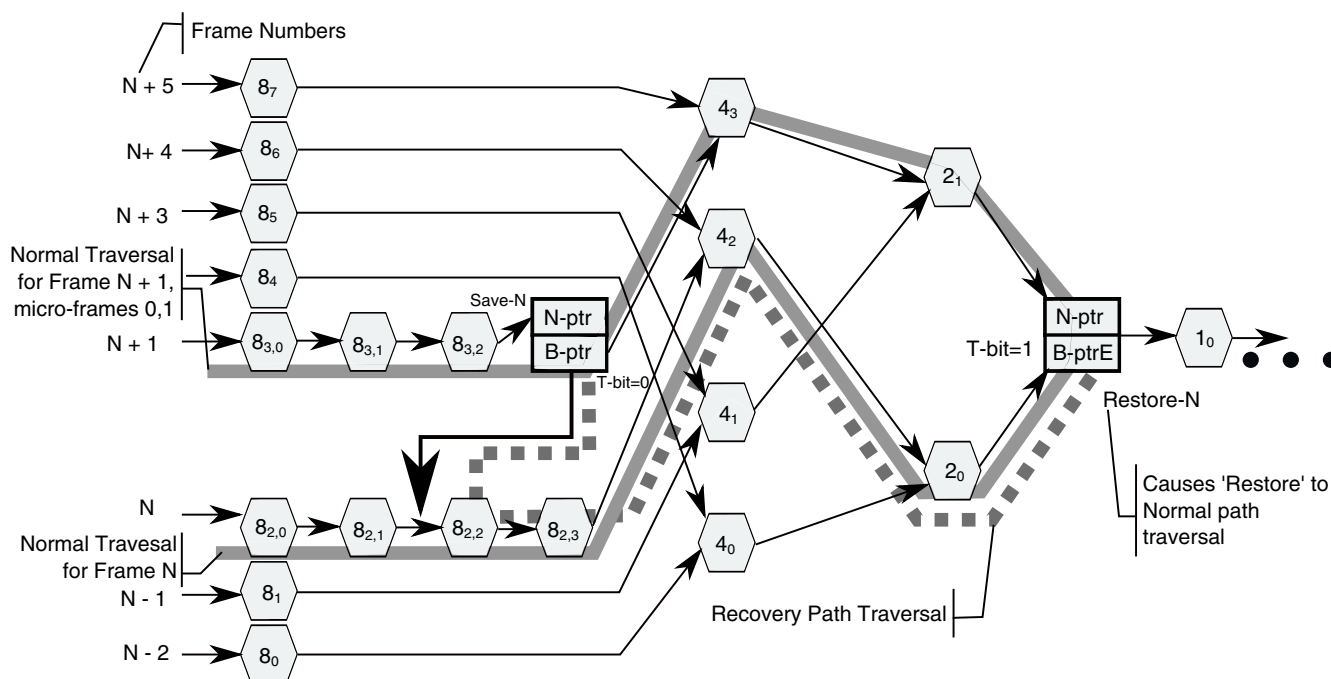
The rules for schedule traversal and limited execution while in *Recovery Path* mode are:

- Always follow the *Normal Path Link Pointer* when it encounters an FSTN that is a *Save-Place* indicator. The host controller must not recursively follow *Save-Place* FSTNs. Therefore, while executing in *Recovery Path* mode, it must never follow an FSTN's *Back Path Link Pointer*.
- Do not process an siTD or, iTD data structure. Simply follow its *Next Link Pointer*.
- Do not process a QH (Queue Head) whose *EPS* field indicates a high-speed device. Simply follow its *Horizontal Link Pointer*.
- When a QH's *EPS* field indicates a Full/Low-speed device, the host controller considers only it for execution if its *SplitXState* is DoComplete (note: this applies whether the *PID Code* indicates an IN or an OUT). See [Execute Transaction](#) and [Tracking Split Transaction Progress for Interrupt Transfers](#) for a complete list of additional conditions that must be met in general for the host controller to issue a bus transaction.
  - The host controller must not execute a Start-split transaction while executing in *Recovery Path* mode. See [Periodic Isochronous - Do Complete Split](#) for special handling when in *Recovery Path* mode.
- Stop traversing the *recovery path* when it encounters an FSTN that is a *Restore* indicator. The host controller unconditionally uses the saved value of the *Save-Place* FSTN's *Normal Path Link Pointer* when returning to the normal path traversal. The

host controller must clear the context of executing a *Recovery Path* when it restores schedule traversal to the *Save-Place* FSTN's *Normal Path Link Pointer*.

- If the host controller determines that there is not enough time left in the micro-frame to complete processing of the periodic schedule, it abandons traversal of the recovery path, and clears the context of executing a recovery path. The result is that at the start of the next consecutive micro-frame, the host controller starts traversal at the frame list.

An example traversal of a periodic schedule that includes FSTNs is illustrated in the following figure.



**Figure 52-23. Example Host Controller Traversal of Recovery Path via FSTNs**

In frame N+1 (micro-frames 0 and 1), when the host controller encounters Save-Path FSTN (Save-N), it observes that Save-N.Back Path Link Pointer.T-bit is zero (definition of a Save-Path indicator). The host controller saves the value of Save-N.Normal Path Link Pointer and follows Save-N.Back Path Link Pointer. At the same time, it sets an internal flag indicating that it is now in *Recovery Path* mode (the recovery path is annotated in the figure above with a large dashed line). The host controller continues traversing data structures on the recovery path and executing only those bus transactions as noted above, on the recovery path until it reaches Restore FSTN (Restore-N). Restore-N.Back Path Link Pointer.T-bit is set to one (definition of a Restore indicator), so the host controller exits *Recovery Path* mode by clearing the internal *Recovery Path* mode flag and commences (restores) schedule traversal using the saved value of the *Save-Place* FSTN's *Normal Path Link Pointer* (for example Save-N.Normal Path Link Pointer). The nodes traversed during these micro-frames include: {83,0, 83,1, 83,2, Save-A, 82,2, 82,3, 42,

$2_0$ , Restore-N,  $4_3$ ,  $2_1$ , Restore-N,  $1_0 \dots$ }. The nodes on the recovery-path are in bold. In frame N (micro-frames 0-7), for this example, the host controller traverses all of the schedule data structures utilizing the *Normal Path Link Pointers* in any FSTNs it encounters. This is because the host controller has not yet encountered a *Save-Place* FSTN so it not executing in *Recovery Path* mode. When it encounters the *Restore* FSTN, (Restore-N), during micro-frames 0 and 1, it uses Restore-N.Normal Path Link Pointer to traverse to the next data structure (that is normal schedule traversal). This is because the host controller must use a Restore FSTN's *Normal Path Link Pointer* when not executing in a *Recovery-Path* mode. The nodes traversed during frame N include:  $\{8_{2,0}$ ,  $8_{2,1}$ ,  $8_{2,2}$ ,  $8_{2,3}$ ,  $4_2$ ,  $2_0$ , Restore-N,  $1_0 \dots$ }.

In frame N+1 (micro-frames 2-7), when the host controller encounters Save-Path FSTN Save-N, it unconditionally follows Save-N.Normal Path Link Pointer. The nodes traversed during these micro-frames include:  $\{8_{3,0}$ ,  $8_{3,1}$ ,  $8_{3,2}$ , Save-A,  $4_3$ ,  $2_1$ , Restore-N,  $1_0 \dots$ }.

### 52.4.3.12.2.3 Software Operational Model for FSTNs

Software must create a consistent, coherent schedule for the host controller to traverse.

When using FSTNs, system software must adhere to the following rules:

- Each *Save-Place* indicator requires a matching *Restore* indicator.
  - The *Save-Place* indicator is an FSTN with a valid *Back Path Link Pointer* and *T-bit* equal to zero.
    - *Back Path Link Pointer.Type* field must be set to indicate the referenced data structure is a queue head. The *Restore* indicator is an FSTN with its *Back Path Link Pointer.T-bit* set to one.
  - A *Restore* FSTN may be matched to one or more *Save-Place* FSTNs. For example, if the schedule includes a poll-rate 1 level, then system software only needs to place a *Restore* FSTN at the beginning of this list in order to match all possible *Save-Place* FSTNs.
- If the schedule does not have elements linked at a poll-rate level of one, and one or more *Save-Place* FSTNs are used, then System Software must ensure the *Restore* FSTN's *Normal Path Link Pointer's T-bit* is set to one, as this is used to mark the end of the periodic list.
- When the schedule does have elements linked at a poll rate level of one, a *Restore* FSTN must be the first data structure on the poll rate one list. All traversal paths from the frame list converge on the poll-rate one list. System software must ensure that *Recovery Path* mode is exited before the host controller is allowed to traverse the poll rate level one list.
- A *Save-Place* FSTN's *Back Path Link Pointer* must reference a queue head data structure. The referenced queue head must be reachable from the previous frame list

location. In other words, if the *Save-Place* FSTN is reachable from frame list offset N, then the FSTN's *Back Path Link Pointer* must reference a queue head that is reachable from frame list offset N-1.

Software should make the schedule as efficient as possible. What this means in this context is that software should have no more than one *Save-Place* FSTN reachable in any single frame. Note there is times when two (or more, depending on the implementation) could exist as full/low-speed footprints change with bandwidth adjustments. This could occur, for example when a bandwidth re-balance causes system software to move the *Save-Place* FSTN from one poll rate level to another. During the transition, software must preserve the integrity of the previous schedule until the new schedule is in place.

#### 52.4.3.12.2.4 Tracking Split Transaction Progress for Interrupt Transfers

To correctly maintain the data stream, the host controller must be able to detect and report errors where data is lost.

For interrupt-IN transfers, data is lost when it makes it into the USB 2.0 hub, but the USB 2.0 host system is unable to get it from the USB 2.0 Hub and into the system before it expires from the transaction translator pipeline.

When a lost data condition is detected, the queue must be halted, thus signaling system software to recover from the error. A data-loss condition exists whenever a start-split is issued, accepted and successfully executed by the USB 2.0 Hub, but the complete-splits get unrecoverable errors on the high-speed link, or the complete-splits do not occur at the correct times. One reason complete-splits might not occur at the right time would be due to host-induced system hold-offs that cause the host controller to miss bus transactions because it cannot get timely access to the schedule in system memory.

The same condition can occur for an interrupt-OUT, but the result is not an endpoint halt condition, but rather effects only the progress of the transfer. The queue head has the following fields to track the progress of each split transaction. These fields are used to keep incremental state about which (and when) portions have been executed.

- *C-prog-mask*. This is an eight-bit bit-vector where the host controller keeps track of which complete-splits have been executed. Due to the nature of the Transaction Translator periodic pipeline, the complete-splits need to be executed in-order. The host controller needs to detect when the complete-splits have not been executed in order. This can only occur due to system hold-offs where the host controller cannot get to the memory-based schedule. *C-prog-mask* is a simple bit-vector that the host controller sets one of the *C-prog-mask* bits for each complete-split executed. The bit position is determined by the micro-frame number in which the complete-split was executed. The host controller always checks *C-prog-mask* before executing a



complete-split transaction. If the previous complete-splits have not been executed then it means one (or more) have been skipped and data has potentially been lost.

- *FrameTag*. This field is used by the host controller during the complete-split portion of the split transaction to tag the queue head with the frame number (*H-Frame* number) when the next complete split must be executed.
- *S-bytes*. This field can be used to store the number of data payload bytes sent during the start-split (if the transaction was an OUT). The *S-bytes* field must be used to accumulate the data payload bytes received during the complete-splits (for an IN).

#### 52.4.3.12.2.5 Split Transaction Execution State Machine for Interrupt

In the following presentation, all references to micro-frame are in the context of a micro-frame within an *H-Frame*.

>As with asynchronous Full- and Low-speed endpoints, a split-transaction state machine is used to manage the split transaction sequence.

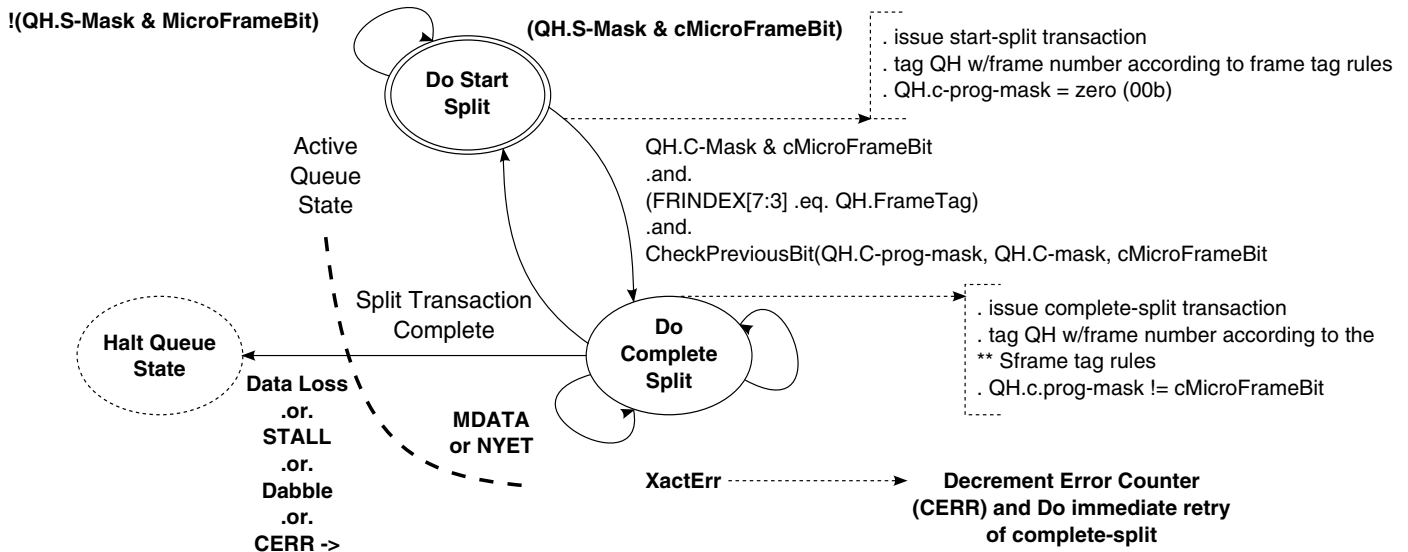
Aside from the fields defined in the queue head for scheduling and tracking the split transaction, the host controller calculates one internal mechanism that is also used to manage the split transaction. The internal calculated mechanism is:

- *cMicroFrameBit*. This is a single-bit encoding of the current micro-frame number. It is an eight-bit value calculated by the host controller at the beginning of every micro-frame. It is calculated from the three least significant bits of the *FRINDEX* register (that is,  $cMicroFrameBit = (1 \text{ shifted-left}(FRINDEX[2:0]))$ ). The *cMicroFrameBit* has at most one bit asserted, which always corresponds to the current micro-frame number. For example, if the current micro-frame is 0, then *cMicroFrameBit* will equal 00000001b. The variable *cMicroFrameBit* is used to compare against the *S-mask* and *C-mask* fields to determine whether the queue head is marked for a start- or complete-split transaction for the current micro-frame.

The following figure illustrates the state machine for managing a complete interrupt split transaction. There are two phases to each split transaction. The first is a single start-split transaction, which occurs when the *SplitXState* is at *Do\_Start* and the single bit in *cMicroFrameBit* has a corresponding bit active in *QH.S-mask*. The transaction translator does not acknowledge the receipt of the periodic start-split, so the host controller unconditionally transitions the state to *Do\_Complete*. Due to the available jitter in the transaction translator pipeline, there will be more than one complete-split transaction scheduled by software for the *Do\_Complete* state. This translates simply to the fact that there are multiple bits set to a one in the *QH.C-mask* field.

The host controller keeps the queue head in the *Do\_Complete* state until the split transaction is complete (see definition below), or an error condition triggers the *three-strikes-rule* (for example, after the host tries the same transaction three times, and each

encounters an error, the host controller will stop retrying the bus transaction and halt the endpoint, thus requiring system software to detect the condition and perform system-dependent recovery).



**Figure 52-24. Split Transaction State Machine for Interrupt**

See Previous Section for the frame tag management rules.

#### Periodic Interrupt - Do Start Split

This is the state software must initialize a full- or low-speed interrupt queue head *StartXState* bit. This state is entered from the Do\_Complete Split state only after the split transaction is complete. This occurs when one of the following events occur: The transaction translator responds to a complete-split transaction with one of the following:

- NAK. A NAK response is a propagation of the full- or low-speed endpoint's NAK response.
- ACK. An ACK response is a propagation of the full- or low-speed endpoint's ACK response. Only occurs on an OUT endpoint.
- DATA 0/1. Only occurs for INs. Indicates that this is the last of the data from the endpoint for this split transaction.
- ERR. The transaction on the low-/full-speed link below the transaction translator had a failure (for example, timeout, bad CRC, etc.).
- NYET (and Last). The host controller issued the last complete-split and the transaction translator responded with a NYET handshake. This means that the start-split was not correctly received by the transaction translator, so it never executed a transaction to the full- or low-speed endpoint, see Section [Periodic Isochronous - Do Complete Split](#) for the definition of 'Last'.

Each time the host controller visits a queue head in this state (once within the Execute Transaction state), it performs the following test to determine whether to execute a start-split.

- *QH.S-mask* is bit-wise anded with *cMicroFrameBit*.

If the result is non-zero, then the host controller will issue a start-split transaction. If the *PIDCode* field indicates an IN transaction, the host controller must zero-out the *QH.S-bytes* field. After the split-transaction has been executed, the host controller sets up state in the queue head to track the progress of the complete-split phase of the split transaction. Specifically, it records the expected frame number into *QH.FrameTag* field (see Section ), set *C-prog-mask* to zero (00h), and exits this state. Note that the host controller must not adjust the value of *CErr* as a result of completion of a start-split transaction.

### Periodic Interrupt - Do Complete Split

This state is entered unconditionally from the Do Start Split state after a start-split transaction is executed on the bus. Each time the host controller visits a queue head in this state (once within the Execute Transaction state), it checks to determine whether a complete-split transaction should be executed now.

There are four tests to determine whether a complete-split transaction should be executed.

- Test A. *cMicroFrameBit* is bit-wise anded with *QH.C-mask* field. A non-zero result indicates that software scheduled a complete-split for this endpoint, during this micro-frame.
- Test B. *QH.FrameTag* is compared with the current contents of *FRINDEX[7:3]*. An equal indicates a match.
- Test C. The complete-split progress bit vector is checked to determine whether the previous bit is set, indicating that the previous complete-split was appropriately executed. An example algorithm for this test is provided below:

```
Algorithm Boolean CheckPreviousBit(QH.C-prog-mask, QH.C-mask, cMicroFrameBit)
Begin
-- Return values:
-- TRUE - no error
-- FALSE - error
--
Boolean rvalue = TRUE;
previousBit = cMicroframeBit logical-rotate-right(1)
-- Bit-wise anding previousBit with C-mask indicates
-- whether there was an intent
-- to send a complete split in the previous micro-frame. So,
-- if the
-- 'previous bit' is set in C-mask, check C-prog-mask to
-- make sure it
-- happened.
If (previousBit bitAND QH.C-mask) then
    If not(previousBit bitAND QH.C-prog-mask) then
        rvalue = FALSE;
    End if
End If
-- If the C-prog-mask already has a one in this bit position,
```

```
-- then an aliasing
-- error has occurred. It will probably get caught by the
-- FrameTag Test, but
-- at any rate it is an error condition that as detectable here
-- should not allow
-- a transaction to be executed.
If (cMicroFrameBit bitAND QH.C-prog-mask) then
    rvalue = FALSE;
End if
return (rvalue)
End Algorithm
```

- Test D. Check to see if a start-split should be executed in this micro-frame. Note this is the same test performed in the Do Start Split state (see Section [Periodic Isochronous - Do Start Split](#)). Whenever it evaluates to TRUE and the controller is NOT processing in the context of a *Recovery Path* mode, it means a start-split should occur in this micro-frame. Test D and Test A evaluating to TRUE at the same time is a system software error. Behavior is undefined.

If (A .and. B .and. C .and. not(D)) then the host controller will execute a complete-split transaction. When the host controller commits to executing the complete-split transaction, it updates *QH.C-prog-mask* by bit-ORing with *cMicroFrameBit*. On completion of the complete-split transaction, the host controller records the result of the transaction in the queue head and sets *QH.FrameTag* to the expected *H-Frame* number (see Section ). The effect to the state of the queue head and thus the state of the transfer depends on the response by the transaction translator to the complete-split transaction. The following responses have the effects (note that any responses that result in decrementing of the *CErr* will result in the queue head being halted by the host controller if the result of the decrement is zero):

- NYET (and Last). On each NYET response, the host controller checks to determine whether this is the last complete-split for this split transaction. Last is defined in this context as the condition where all of the scheduled complete-splits have been executed. If it is the last complete-split (with a NYET response), then the transfer state of the queue head is not advanced (never received any data) and this state exited. The transaction translator must have responded to all the complete-splits with NYETs, meaning that the start-split issued by the host controller was not received. The start-split should be retried at the next poll period.
- The test for whether this is the Last complete split can be performed by XOR *QH.C-mask* with *QH.C-prog-mask*. If the result is all zeros then all complete-splits have been executed. When this condition occurs, the *XactErr* status bit is set to a one and the *CErr* field is decremented.
- NYET (and not Last). See above description for testing for Last. The complete-split transaction received a NYET response from the transaction translator. Do not update any transfer state (except for *C-prog-mask* and *FrameTag*) and stay in this state. The host controller must not adjust *CErr* on this response.

- Transaction Error (XactErr). Timeout, data CRC failure, etc. The *CErr* field is decremented and the *XactErr* bit in the *Status* field is set to a one. The complete split transaction is *immediately* retried (if *CErr* is non-zero). If there is not enough time in the micro-frame to complete the retry and the endpoint is an IN, or *CErr* is decremented to a zero from a one, the queue is halted. If there is not enough time in the micro-frame to complete the retry and the endpoint is an OUT and *CErr* is not zero, then this state is exited (that is, return to Do Start Split). This results in a retry of the entire OUT split transaction, at the next poll period. Refer to Chapter 11 Hubs (specifically the section full- and low-speed Interrupts) in the USB Specification Revision 2.0 for detailed requirements on why these errors must be immediately retried.
- ACK. This can only occur if the target endpoint is an OUT. The target endpoint ACK'd the data and this response is a propagation of the endpoint ACK up to the host controller. The host controller must advance the state of the transfer. The *Current Offset* field is incremented by *Maximum Packet Length* or *Bytes to Transfer*, whichever is less. The field *Bytes To Transfer* is decremented by the same amount. And the data toggle bit (*dt*) is toggled. The host controller will then exit this state for this queue head. The host controller must reload *CErr* with maximum value on this response. Advancing the transfer state may cause other process events such as retirement of the qTD and advancement of the queue (see Section [Managing Control/Bulk/Interrupt Transfers through Queue Heads](#)).
- MDATA. This response will only occur for an IN endpoint. The transaction translator responded with zero or more bytes of data and an MDATA PID. The incremental number of bytes received is accumulated in *QH.S-bytes*. The host controller must not adjust *CErr* on this response.
- DATA0/1. This response may only occur for an IN endpoint. The number of bytes received is added to the accumulated byte count in *QH.S-bytes*. The state of the transfer is advanced by the result and the host controller will exit this state for this queue head.
- Advancing the transfer state may cause other processing events such as retirement of the qTD and advancement of the queue (see Section [Managing Control/Bulk/Interrupt Transfers through Queue Heads](#)).
- If the data sequence PID does not match the expected, the entirety of the data received in this split transaction is ignored, the transfer state is not advanced and this state is exited.
- NAK. The target endpoint Nak'd the full- or low-speed transaction. The state of the transfer is not advanced, and this state is exited. The host controller must reload *CErr* with maximum value on this response.

- **ERR.** There was an error during the full- or low-speed transaction. The ERR status bit is set to a one, *Cerr* is decremented, the state of the transfer is not advanced, and this state is exited.
- **STALL.** The queue is halted (an exit condition of the Execute Transaction state). The status field bits: *Active* bit is set to zero and the *Halted* bit is set to a one and the qTD is retired. Responses which are not enumerated in the list or which are received out of sequence are illegal and may result in undefined host controller behavior. The other possible combinations of tests A, B, C, and D may indicate that data or response was lost. The table below lists the possible combinations and the appropriate action.

**Table 52-47. Interrupt IN/OUT Do Complete Split State Execution Criteria**

Condition	Action	Description
not(A) not(D)	Ignore QHD	Neither a start nor complete-split is scheduled for the current micro-frame. Host controller should continue walking the schedule.
A not(C)	If PIDCode = IN Halt QHD If PIDCode = OUT Retry start-split	Progress bit check failed. These means a complete-split has been missed. There is the possibility of lost data. If <i>PIDCode</i> is an IN, then the Queue head must be halted.  If <i>PIDCode</i> is an OUT, then the transfer state is not advanced and the state exited (for example, start-split is retried). This is a host-induced error and does not effect <i>CERR</i> .  In either case, set the <i>Missed Micro-frame</i> bit in the status field to a one.
A not(B) C	If PIDCode = IN Halt QHD If PIDCode = OUT Retry start-split	<i>QH.FrameTag</i> test failed. This means that exactly one or more <i>H-Frames</i> have been skipped. This means complete-splits and have missed. There is the possibility of lost data. If <i>PIDCode</i> is an IN, then the Queue head must be halted.  If <i>PIDCode</i> is an OUT, then the transfer state is not advanced and the state exited (for example, start-split is retried). This is a host-induced error and does not effect <i>CERR</i> .  In either case, set the <i>Missed Micro-frame</i> bit in the status field to a one.
A B C not(D)	Execute complete-split	This is the non-error case where the host controller executes a complete-split transaction.

Table continues on the next page...

**Table 52-47. Interrupt IN/OUT Do Complete Split State Execution Criteria (continued)**

D	If <i>PIDCode</i> = IN Halt QHD If <i>PIDCode</i> = OUT Retry start-split	<p>This is a degenerate case where the start-split was issued, but all of the complete-splits were skipped and all possible intervening opportunities to detect the missed data failed to fire. If <i>PIDCode</i> is an IN, then the Queue head must be halted.</p> <p>If <i>PIDCode</i> is an OUT, then the transfer state is not advanced and the state exited (for example, start-split is retried). This is a host-induced error and does not effect <i>CERR</i>.</p> <p>In either case, set the <i>Missed Micro-frame</i> bit in the status field to a one. Note: When executing in the context of a <i>Recovery Path</i> mode, the host controller is allowed to process the queue head and take the actions indicated above, or it may wait until the queue head is visited in the normal processing mode. Regardless, the host controller must not execute a start-split in the context of a executing in a <i>Recovery Path</i> mode.</p>
---	--	--

### Managing QH.FrameTag Field

The *QH.FrameTag* field in a queue head is completely managed by the host controller. The rules for setting *QH.FrameTag* are simple:

- Rule 1: If transitioning from Do Start Split to Do Complete Split and the current value of *FRINDEX*[2:0] is 6 *QH.FrameTag* is set to *FRINDEX*[7:3] + 1. This accommodates split transactions whose start-split and complete-splits are in different *H-Frames* (case 2a, see [Figure 52-21](#)).
- Rule 2: If the current value of *FRINDEX*[2:0] is 7, *QH.FrameTag* is set to *FRINDEX*[7:3] + 1. This accommodates staying in Do Complete Split for cases 2a, 2b, and 2c ([Figure 52-21](#)).
- Rule 3: If transitioning from Do\_Start Split to Do Complete Split and the current value of *FRINDEX*[2:0] is not 6, or currently in Do Complete Split and the current value of (*FRINDEX*[2:0]) is not 7, *FrameTag* is set to *FRINDEX*[7:3]. This accommodates all other cases ([Figure 52-21](#)).

#### 52.4.3.12.2.6 Rebalancing the periodic schedule

System software must occasionally adjust a periodic queue head's S-mask and C-mask fields during operation.

This need occurs when adjustments to the periodic schedule create a new bandwidth budget and one or more queue head's are assigned new execution footprints (that is, new S-mask and C-mask values).

It is imperative that System software must not update these masks to new values in the midst of a split transaction. In order to avoid any race conditions with the update, the EHCI host controller provides a simple assist to system software. System software sets the *Inactivate-on-next-Transaction* (*I*) bit to a one to signal the host controller that it



intends to update the S-mask and C-mask on this queue head. System software will then wait for the host controller to observe the *I-bit* is a one and transition the *Active* bit to a zero. The rules for how and when the host controller sets the *Active* bit to zero are enumerated below:

- If the *Active* bit is a zero, no action is taken. The host controller does not attempt to advance the queue when the *I-bit* is a one.
- If the *Active* bit is a one and the *SplitXState* is DoStart (regardless of the value of *S-mask*), the host controller will simply set *Active* bit to a zero. The host controller is not required to write the transfer state back to the *current* qTD. Note that if the *S-mask* indicates that a start-split is scheduled for the current micro-frame, the host controller must not issue the start-split bus transaction. It must set the *Active* bit to zero.

System software must save transfer state before setting the *I-bit* to a one. This is required so that it can correctly determine what transfer progress (if any) occurred after the *I-bit* was set to a one and the host controller executed its final bus-transaction and set *Active* to a zero.

After system software has updated the S-mask and C-mask, it must then reactivate the queue head. Because the *Active* bit and the *I-bit* cannot be updated with the same write, system software needs to use the following algorithm to coherently re-activate a queue head that has been stopped via the *I-bit*.

1. Set the *Halted* bit to a one, then
2. Set the *I-bit* to a zero, then
3. Set the *Active* bit to a one and the *Halted* bit to a zero in the same write.

Setting the *Halted* bit to a one inhibits the host controller from attempting to advance the queue between the time the *I-bit* goes to a zero and the *Active* bit goes to a one.

### 52.4.3.12.3 Split Transaction Isochronous

Full-speed isochronous transfers are managed using the split-transaction protocol through a USB 2.0 transaction translator in a USB2.0 Hub. The EHCI controller utilizes siTD data structure to support the special requirements of isochronous split-transactions.

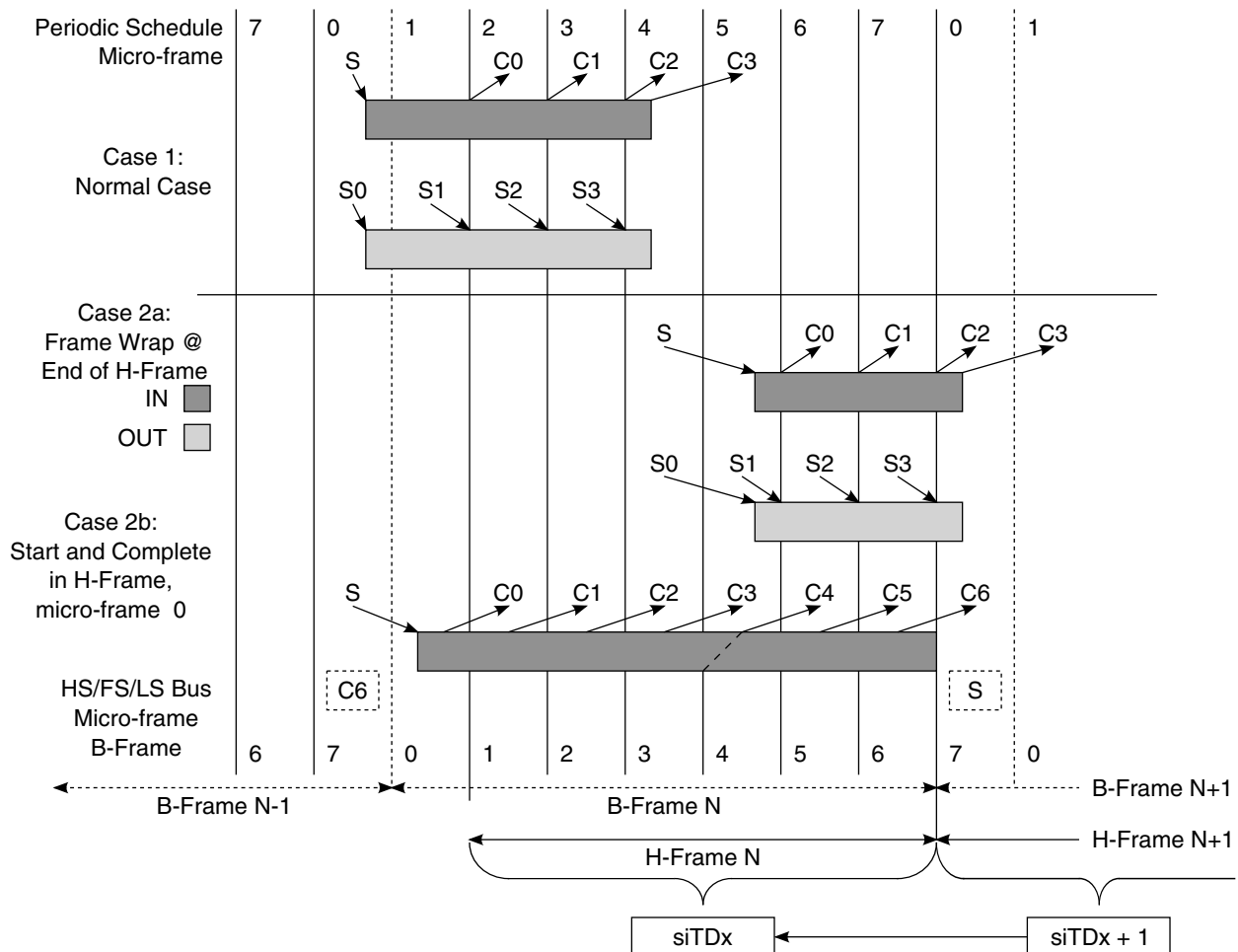
This data structure uses the scheduling model of isochronous TDs (iTDD, Section [Isochronous \(High-Speed\) Transfer Descriptor \(iTDD\)](#)) (see Section [Managing Isochronous Transfers Using iTDDs](#) for the operational model of iTDDs) with the contiguous data feature provided by queue heads. This simple arrangement allows a single isochronous scheduling model and adds the additional feature that all data received from the endpoint (per split transaction) must land into a contiguous buffer.



### 52.4.3.12.3.1 Split Transaction Scheduling Mechanisms for Isochronous

Full-speed isochronous transactions are managed through a transaction translator's periodic pipeline. As with full- and low-speed interrupt, system software manages each transaction translator's periodic pipeline by budgeting and scheduling exactly during which micro-frames the start-splits and complete-splits for each full-speed isochronous endpoint occur.

The requirements described in Section [Split Transaction Scheduling Mechanisms for Interrupt](#) apply. The following figure illustrates the general scheduling boundary conditions that are supported by the EHCI periodic schedule. The  $SX$  and  $CX$  labels indicate micro-frames where software can schedule start- and complete-splits (respectively). The *H-Frame* boundaries are marked with a large, solid bold vertical line. The *B-Frame* boundaries are marked with a large, bold, dashed line. The bottom of the figure illustrates the relationship of an siTD to the *H-Frame*.



**Figure 52-25. Split Transaction, Isochronous Scheduling Boundary Conditions**

When the endpoint is an isochronous OUT, there are only start-splits, and no complete-splits. When the endpoint is an isochronous IN, there is at most one start-split and one to  $N$  complete-splits. The scheduling boundary cases are:

- *Case 1:* The entire split transaction is completely bounded by an *H-Frame*. For example: the start-splits and complete-splits are all scheduled to occur in the same *H-Frame*.
- *Case 2a:* This boundary case is where one or more (at most two) complete-splits of a split transaction IN are scheduled across an *H-Frame* boundary. This can only occur when the split transaction has the possibility of moving data in *B-Frame*, micro-frames 6 or 7 (*H-Frame* micro-frame 7 or 0). When an *H-Frame* boundary wrap condition occurs, the scheduling of the split transaction spans more than one location in the periodic list. (For example, it takes two siTDs in adjacent periodic frame list locations to fully describe the scheduling for the split transaction.)
- Although the scheduling of the split transaction may take two data structures, all of the complete-splits for each full-speed IN isochronous transaction must use only one data pointer. For this reason, siTDs contain a back pointer, the use of which is described below.
- Software must never schedule full-speed isochronous OUTs across an *H-Frame* boundary.
- *Case 2b:* This case can only occur for a very large isochronous IN. It is the only allowed scenario where a start-split and complete-split for the same endpoint can occur in the same micro-frame. Software must enforce this rule by scheduling the large transaction first. Large is defined to be anything larger than 579 byte maximum packet size.

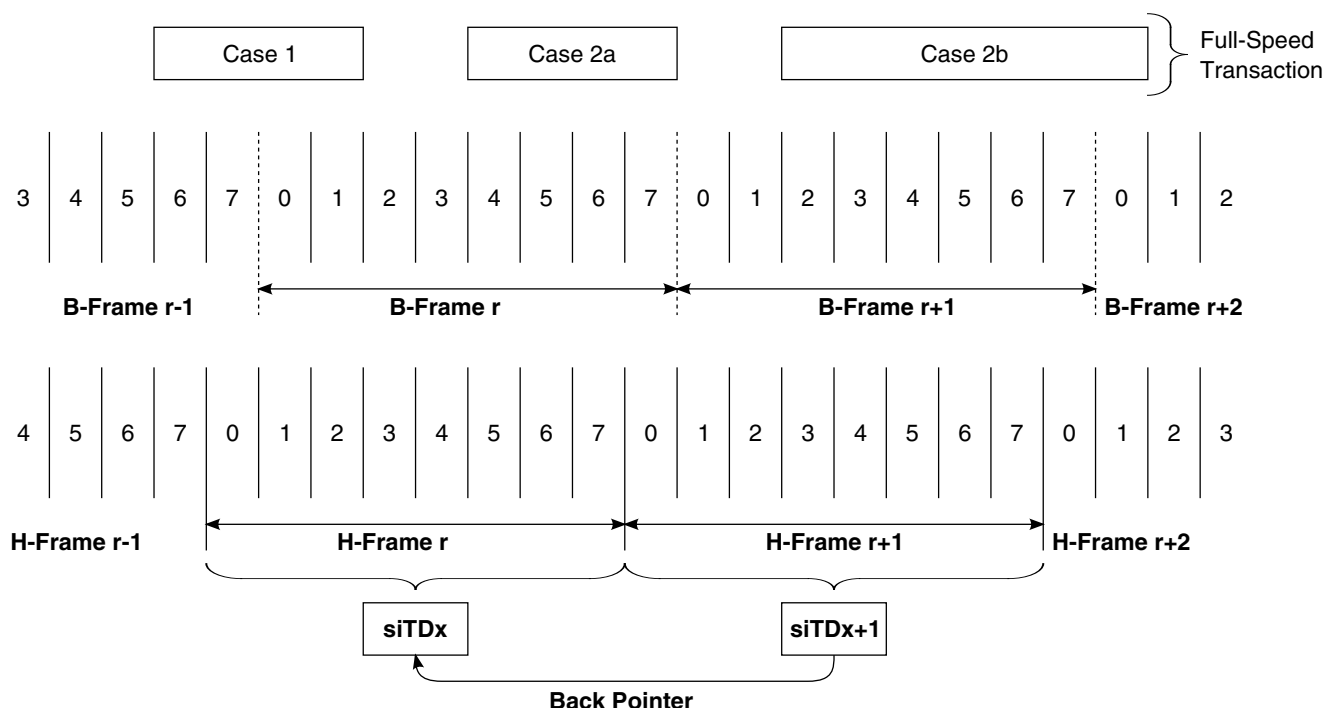
A subset of the same mechanisms employed by full- and low-speed interrupt queue heads are employed in siTDs to schedule and track the portions of isochronous split transactions. The following fields are initialized by system software to instruct the host controller when to execute portions of the split transaction protocol.

- *SplitXState*. This is a single bit residing in the *Status* field of an siTD (see [Figure 52-26](#)). This bit is used to track the current state of the split transaction. The rules for managing this bit are described in [Section Split Transaction Execution State Machine for Interrupt](#).
- *Frame S-mask*. This is a bit-field where-in system software sets a bit corresponding to the micro-frame (within an *H-Frame*) that the host controller should execute a start-split transaction. This is always qualified by the value of the *SplitXState* bit. For example, referring to the IN example in [Figure 52-25](#), case one, the *S-mask* would have a value of 00000001b indicating that if the siTD is traversed by the host controller, and the *SplitXState* indicates Do Start Split, and the current micro-frame as indicated by USB\_n\_FRINDEX[2:0] is 0, then execute a start-split transaction.

- *Frame C-mask*. This is a bit-field where system software sets one or more bits corresponding to the micro-frames (within an *H-Frame*) that the host controller should execute complete-split transactions. The interpretation of this field is always qualified by the value of the *SplitXState* bit. For example, referring to the IN example in [Figure 52-25](#), case one, the *C-mask* would have a value of 00111100b indicating that if the siTD is traversed by the host controller, and the *SplitXState* indicates Do Complete Split, and the current micro-frame as indicated by *USB\_n\_FRINDEX*[2:0] is 2, 3, 4, or 5, then execute a complete-split transaction.
- *Back Pointer*. This field in a siTD is used to complete an IN split-transaction using the previous *H-Frame*'s siTD. This is only used when the scheduling of the complete-splits span an *H-Frame* boundary.

There exists a one-to-one relationship between a high-speed isochronous split transaction (including all start- and complete-splits) and one full-speed isochronous transaction. An siTD contains (amongst other things) buffer state and split transaction scheduling information. An siTD's buffer state always maps to one full-speed isochronous data payload. This means that for any full-speed transaction payload, a single siTD's data buffer must be used. This rule applies to both IN and OUTs. An siTD's scheduling information usually also maps to one high-speed isochronous split transaction. The exception to this rule is the *H-Frame* boundary wrap cases mentioned above.

The siTD data structure describes at most, one frame's worth of high-speed transactions and that description is strictly bounded within a frame boundary. The figure below illustrates some examples. On the top are examples of the full-speed transaction footprints for the boundary scheduling cases described above. In the middle are time-frame references for both the *B-Frames* (HS/FS/LS Bus) and the *H-Frames*. On the bottom is illustrated the relationship between the scope of an siTD description and the time references. Each *H-Frame* corresponds to a single location in the periodic frame list. The implication is that each siTD is reachable from a single periodic frame list location at a time.



**Figure 52-26. siTD Scheduling Boundary Examples**

Each case is described below:

- *Case 1:* One siTD is sufficient to describe and complete the isochronous split transaction because the whole isochronous split transaction is tightly contained within a single *H-Frame*.
- *Case 2a, 2b:* Although both INs and OUTs can have these footprints, OUTs always take only one siTD to schedule. However, INs (for these boundary cases) require two siTDs to complete the scheduling of the isochronous split transaction. siTD<sub>x</sub> is used to always issue the start-split and the first *N* complete-splits. The full-speed transaction (for these cases) can deliver data on the full-speed bus segment during micro-frame 7 of *H-Frame*<sub>Y+1</sub>, or micro-frame 0 of *H-Frame*<sub>Y+2</sub>. The complete splits are scheduled using siTD<sub>X+2</sub> (not shown). The complete-splits to extract this data must use the buffer pointer from siTD<sub>X+1</sub>. The only way for the host controller to reach siTD<sub>X+1</sub> from *H-Frame*<sub>Y+2</sub> is to use siTD<sub>X+2</sub>'s back pointer. The host controller rules for when to use the back pointer are described in Section [Periodic Isochronous - Do Complete Split](#).

Software must apply the following rules when calculating the schedule and linking the schedule data structures into the periodic schedule:

- Software must ensure that an isochronous split-transaction is started so that it will complete before the end of the *B-Frame*.
- Software must ensure that for a single full-speed isochronous endpoint, there is never a start-split and complete-split in *H-Frame, micro-frame 1*. This is mandated as a rule so that case 2a and case 2b can be discriminated. According to the core USB specification, the long isochronous transaction illustrated in Case 2b, could be scheduled so that the start-split was in micro-frame 1 of *H-Frame N* and the last complete-split would need to occur in micro-frame 1 of *H-Frame N+1*. However, it is impossible to discriminate between cases 2a and case 2b, which has significant impact on the complexity of the host controller.

### 52.4.3.12.3.2 Tracking Split Transaction Progress for Isochronous Transfers

To correctly maintain the data stream, the host controller must be able to detect and report errors where device to host data is lost. Isochronous endpoints do not employ the concept of a halt on error, however the host is required to identify and report per-packet errors observed in the data stream. This includes schedule traversal problems (skipped micro-frames), timeouts and corrupted data received.

In similar kind to interrupt split-transactions, the portions of the split transaction protocol must execute in the micro-frames they are scheduled. The queue head data structure used to manage full- and low-speed interrupt has several mechanisms for tracking when portions of a transaction have occurred. Isochronous transfers use siTDs, for their transfers, and the data structures are only reachable via the schedule in the exact micro-frame in which they are required (so all the mechanism employed for tracking in queue heads is not required for siTDs). Software has the option of reusing siTD several times in the complete periodic schedule. However, it must ensure that the results of split transaction *N* are consumed and the siTD reinitialized (activated) before the host controller gets back to the siTD (in a future micro-frame).

Split-transaction isochronous OUTs utilize a low-level protocol to indicate which portions of the split transaction data have arrived. Control over the low-level protocol is exposed in an siTD via the fields *Transaction Position (TP)* and *Transaction Count (T-count)*. If the entire data payload for the OUT split transaction is larger than 188 bytes, there will be more than one start-split transaction, each of which require proper annotation. If host hold-offs occur, then the sequence of annotations received from the host will not be complete, which is detected and handled by the transaction translator. See Section [Periodic Isochronous - Do Start Split](#) for a description on how these fields are used during a sequence of start-split transactions.

The fields *siTD.T-Count* and *siTD.TP* are used by the host controller to drive and sequence the transaction position annotations. It is the responsibility of system software to properly initialize these fields in each siTD. Once the budget for a split-transaction

isochronous endpoint is established, *S-mask*, *T-Count*, and *TP* initialization values for all the siTD associated with the endpoint are constant. They remain constant until the budget for the endpoint is recalculated by software and the periodic schedule adjusted.

For IN-endpoints, the transaction translator simply annotates the response data packets with enough information to allow the host controller to identify the last data. As with split transaction Interrupt, it is the host controller's responsibility to detect when it has missed an opportunity to execute a complete-split. The following field in the siTD is used to track and detect errors in the execution of a split transaction for an IN isochronous endpoint.

- *C-prog-mask*. This is an eight-bit bit-vector where the host controller keeps track of which complete-splits have been executed. Due to the nature of the Transaction Translator periodic pipeline, the complete-splits need to be executed in-order. The host controller needs to detect when the complete-splits have not been executed in order. This can only occur due to system hold-offs where the host controller cannot get to the memory-based schedule. *C-prog-mask* is a simple bit-vector that the host controller sets a bit for each complete-split executed. The bit position is determined by the micro-frame (USB\_n\_FRINDEX[2:0]) number in which the complete-split was executed. The host controller always checks *C-prog-mask* before executing a complete-split transaction. If the previous complete-splits have not been executed, then it means one (or more) have been skipped and data has potentially been lost. System software is required to initialize this field to zero before setting an siTD's *Active* bit to a one.

If a transaction translator returns with the final data before all of the complete-splits have been executed, the state of the transfer is advanced so that the remaining complete-splits are not executed. Refer to Section [Asynchronous - Do Complete Split](#) for a description on how the state of the transfer is advanced. It is important to note that an IN siTD is retired based solely on the responses from the Transaction Translator to the complete-split transactions. This means, for example, that it is possible for a transaction translator to respond to a complete-split with an MDATA PID. The number of bytes in the MDATA's data payload could cause the siTD field *Total Bytes to Transfer* to decrement to zero. This response can occur, before all of the scheduled complete-splits have been executed. In other interface, data structures (for example, high-speed data streams through queue heads), the transition of *Total Bytes to Transfer* to zero signals the end of the transfer and results in setting of the *Active* bit to zero. However, in this case, the result has not been delivered by the Transaction Translator and the host must continue with the next complete-split transaction to extract the residual transaction state. This scenario occurs because of the pipeline rules for a Transaction Translator (see Chapter 11 of the Universal Serial Bus Revision 2.0). In summary the periodic pipeline rules require that on a micro-frame boundary, the Transaction Translator will hold the final two bytes received (if it has not seen an End Of Packet (EOP)) in the full-speed bus pipe stage and give the



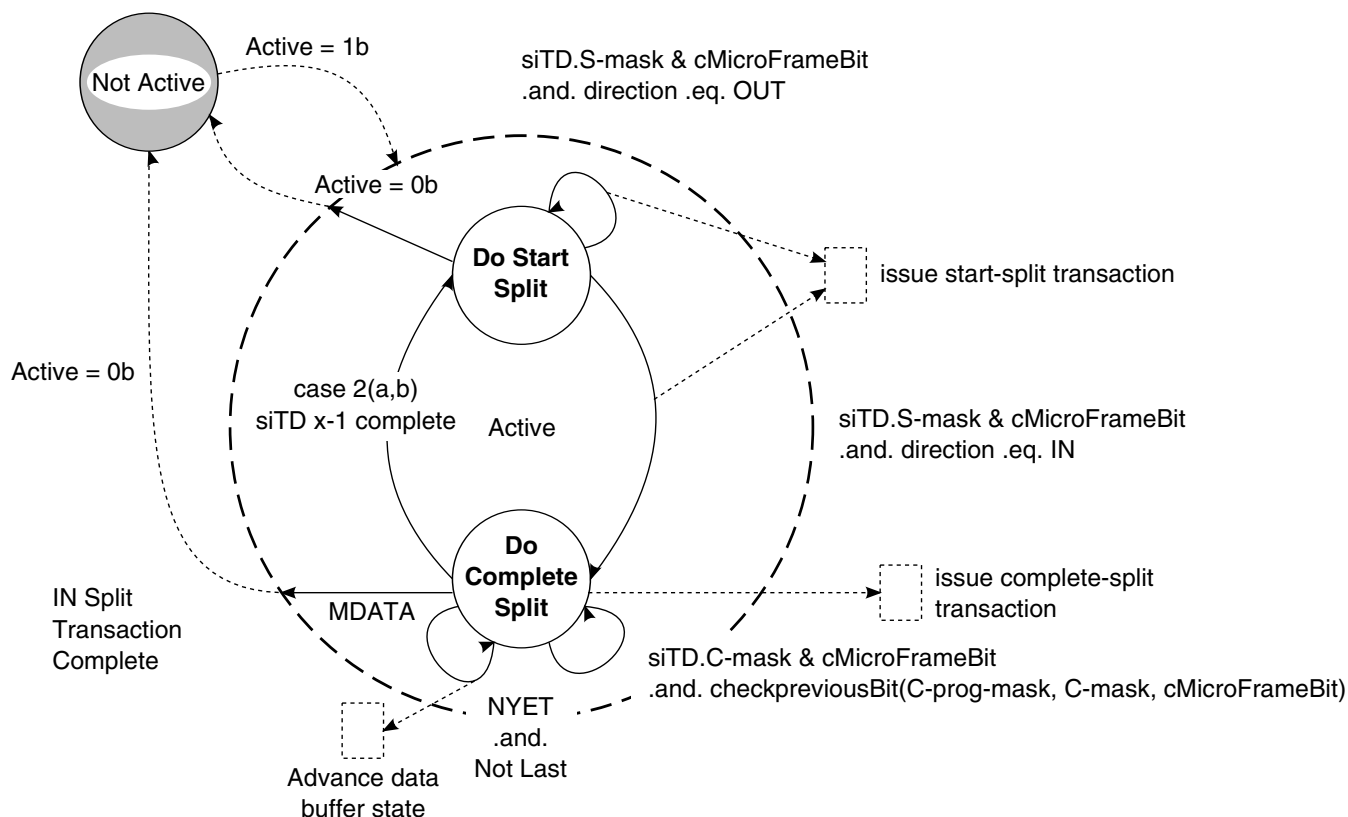
remaining bytes to the high-speed pipeline stage. At the micro-frame boundary, the Transaction Translator could have received the entire packet (including both CRC bytes) but not received the packet EOP. In the next micro-frame, the Transaction Translator will respond with an MDATA and send all of the data bytes (with the two CRC bytes being held in the full-speed pipeline stage). This could cause the siTD to decrement its *Total Bytes to Transfer* field to zero, indicating it has received all expected data. The host must still execute one more (scheduled) complete-split transaction in order to extract the results of the full-speed transaction from the Transaction Translator (for example, the Transaction Translator may have detected a CRC failure, and this result must be forwarded to the host).

If the host experiences hold-offs that cause the host controller to skip one or more (but not all) scheduled split transactions for an isochronous OUT, then the protocol to the transaction translator will not be consistent and the transaction translator will detect and react to the problem. Likewise, for host hold-offs that cause the host controller to skip one or more (but not all) scheduled split transactions for an isochronous IN, the *C-prog-mask* is used by the host controller to detect errors. However, if the host experiences a hold-off that causes it to skip all of an siTD, or an siTD expires during a host hold off (for example, a hold-off occurs and the siTD is no longer reachable by the host controller in order for it to report the hold-off event), then system software must detect that the siTDs have not been processed by the host controller (that is, state not advanced) and report the appropriate error to the client driver.

### 52.4.3.12.3.3 Split Transaction Execution State Machine for Isochronous

In the following presentation, all references to micro-frame are in the context of a micro-frame within an *H-Frame*.

If the *Active* bit in the *Status* byte is a zero, the host controller will ignore the siTD and continue traversing the periodic schedule. Otherwise the host controller will process the siTD as specified below. A split transaction state machine is used to manage the split-transaction protocol sequence. The host controller uses the fields defined in Section [Tracking Split Transaction Progress for Interrupt Transfers](#), plus the variable *cMicroFrameBit* defined in Section [Split Transaction Execution State Machine for Interrupt](#) to track the progress of an isochronous split transaction. The figure below illustrates the state machine for managing an siTD through an isochronous split transaction. Bold, dotted circles denote the state of the *Active* bit in the *Status* field of a siTD. The Bold, dotted arcs denote the transitions between these states. Solid circles denote the states of the split transaction state machine and the solid arcs denote the transitions between these states. Dotted arcs and boxes reference actions that take place either as a result of a transition or from being in a state.



**Figure 52-27. Split Transaction State Machine for Isochronous**

#### 52.4.3.12.3.4 Periodic Isochronous - Do Start Split

Isochronous split transaction OUTs use only this state.

An siTD for a split-transaction isochronous IN is either initialized to this state, or the siTD transitions to this state from Do Complete Split when a case 2a (IN) or 2b scheduling boundary isochronous split-transaction completes.

Each time the host controller reaches an active siTD in this state, it checks the *siTD.S-mask* against *cMicroFrameBit*. If there is a one in the appropriate position, the siTD will execute a start-split transaction. By definition, the host controller cannot *reach* an siTD at the wrong time. If the *I/O* field indicates an IN, then the start-split transaction includes only the extended token plus the full-speed token. Software must initialize the *siTD.Total Bytes To Transfer* field to the number of bytes expected. This is usually the maximum packet size for the full-speed endpoint. The host controller exits this state when the start-split transaction is complete.

The remainder of this section is specific to an isochronous OUT endpoint (that is, the *I/O* field indicates an OUT). When the host controller executes a start-split transaction for an isochronous OUT it includes a data payload in the start-split transaction. The memory



buffer address for the data payload is constructed by concatenating *siTD.Current Offset* with the page pointer indicated by the page selector field (*siTD.P*). A zero in this field selects Page 0 and a 1 selects Page 1. During the start-split for an OUT, if the data transfer crosses a page boundary during the transaction, the host controller must detect the page cross, update the *siTD.P*-bit from a zero to a one, and begin using the *siTD.Page 1* with *siTD.Current Offset* as the memory address pointer. The field *siTD.TP* is used to annotate each start-split transaction with the indication of which part of the split-transaction data the current payload represents (ALL, BEGIN, MID, END). In all cases the host controller simply uses the value in *siTD.TP* to mark the start-split with the correct transaction position code.

*T-Count* is always initialized to the number of start-splits for the current frame. *TP* is always initialized to the first required transaction position identifier. The scheduling boundary case (see [Figure 52-26](#)) is used to determine the initial value of *TP*. The initial cases are summarized in the following table.

**Table 52-48. Initial Conditions for OUT siTD's TP and T-count Fields**

Case	T-count	TP	Description
1, 2a	=1	ALL	When the OUT data payload is less than (or equal to) 188 bytes, only one start-split is required to move the data. The one start-split must be marked with an ALL.
1, 2a	!=1	BEGIN	When the OUT data payload is greater than 188 bytes more than one start-split must be used to move the data. The initial start-split must be marked with a BEGIN.

After each start-split transaction is complete, the host controller updates *T-Count* and *TP* appropriately so that the next start-split is correctly annotated.

The table below illustrates all of the *TP* and *T-count* transitions, which must be accomplished by the host controller.

**Table 52-49. Transaction Position (TP)/Transaction Count (T-Count) Transition Table**

TP	T-count next	TP next	Description
ALL	0	N/A	Transition from ALL, to done.
BEGIN	1	END	Transition from BEGIN to END. Occurs when <i>T-count</i> starts at 2.
BEGIN	!=1	MID	Transition from BEGIN to MID. Occurs when <i>T-count</i> starts at greater than 2.
MID	!=1	MID	<i>TP</i> stays at MID while <i>T-count</i> is not equal to 1 (that is, greater than 1). This case can occur for any of the scheduling boundary cases where the <i>T-count</i> starts greater than 3.
MID	1	END	Transition from MID to END. This case can occur for any of the scheduling boundary cases where the <i>T-count</i> starts greater than 2.

The start-split transactions do not receive a handshake from the transaction translator, so the host controller always advances the transfer state in the siTD after the bus transaction is complete. To advance the transfer state the following operations take place:

- The *siTD.Total Bytes To Transfer* and the *siTD.Current Offset* fields are adjusted to reflect the number of bytes transferred.
- The *siTD.P* (page selector) bit is updated appropriately.
- The *siTD.TP* and *siTD.T-count* fields are updated appropriately as defined in [Table 52-49](#).

These fields are then written back to the memory based siTD. The *S-mask* is fixed for the life of the current budget. As mentioned above, *TP* and *T-count* are set specifically in each siTD to reflect the data to be sent from this siTD. Therefore, regardless of the value of *S-mask*, the actual number of start-split transactions depends on *T-count* (or equivalently, *Total Bytes to Transfer*). The host controller must set the *Active* bit to a zero when it detects that all of the schedule data has been sent to the bus. The preferred method is to detect when *T-Count* decrements to zero as a result of a start-split bus transaction. Equivalently, the host controller can detect when *Total Bytes to Transfer* decrements to zero. Either implementation must ensure that if the initial condition is *Total Bytes to Transfer* equal to zero and *T-count* is equal to a one, then the host controller will issue a single start-split, with a zero-length data payload. Software must ensure that *TP*, *T-count* and *Total Bytes to Transfer* are set to deliver the appropriate number of bus transactions from each siTD. An inconsistent combination will yield undefined behavior.

If the host experiences hold-offs that cause the host controller to skip start-split transactions for an OUT transfer, the state of the transfer will not progress appropriately. The transaction translator will observe protocol violations in the arrival of the start-splits for the OUT endpoint (that is, the transaction position annotation will be incorrect as received by the transaction translator).

Example scenarios are described in [Section Split Transaction for Isochronous - Processing Examples](#).

A host controller implementation can optionally track the progress of an OUT split transaction by setting appropriate bits in the *siTD.C-prog-mask* as it executes each scheduled start-split. The *checkPreviousBit()* algorithm defined in [Periodic Isochronous - Do Complete Split](#) can be used prior to executing each start-split to determine whether start-splits were skipped. The host controller can use this mechanism to detect missed micro-frames. It can then set the siTD's *Active* bit to zero and stop execution of this siTD. This saves on both memory and high-speed bus bandwidth.

### 52.4.3.12.3.5 Periodic Isochronous - Do Complete Split

This state is only used by a split-transaction isochronous IN endpoint.

This state is entered unconditionally from the Do Start State after a start-split transaction is executed for an IN endpoint. Each time the host controller visits an siTD in this state, it conducts a number of tests to determine whether it should execute a complete-split transaction. The individual tests are listed below. The sequence they are applied depends on which micro-frame the host controller is currently executing which means that the tests might not be applied until after the siTD referenced from the back pointer has been fetched.

- Test A. *cMicroFrameBit* is bit-wise anded with *siTD.C-mask* field. A non-zero result indicates that software scheduled a complete-split for this endpoint, during this micro-frame. This test is always applied to a newly fetched siTD that is in this state.
- Test B. The *siTD.C-prog-mask* bit vector is checked to determine whether the previous complete splits have been executed. An example algorithm is below (this is slightly different than the algorithm used in Section [Periodic Isochronous - Do Complete Split](#)). The sequence in which this test is applied depends on the current value of *USB\_n\_FRINDEX[2:0]*. If *USB\_n\_FRINDEX[2:0]* is 0 or 1, it is not applied until the back pointer has been used. Otherwise it is applied immediately.

Algorithm Boolean CheckPreviousBit(*siTD.C-prog-mask*, *siTD.C-mask*, *cMicroFrameBit*)

Begin

```

Boolean rvalue = TRUE;
previousBit = cMicroFrameBit rotate-right(1)
-- Bit-wise anding previousBit with C-mask indicates whether there was an intent
-- to send a complete split in the previous micro-frame. So, if the
-- 'previous bit' is set in C-mask, check C-prog-mask to make sure it
-- happened.
if previousBit bitAND siTD.C-mask then
    if not (previousBit bitAND siTD.C-prog-mask) then
        rvalue = FALSE
    End if
End if
Return rvalue

```

End Algorithm

If Test A is true and *USB\_n\_FRINDEX[2:0]* is zero or one, then this is a case 2a or 2b scheduling boundary (see [Figure 52-25](#)). See Section [Periodic Isochronous - Do Complete Split](#) for details in handling this condition.

If Test A and Test B evaluate to true, then the host controller will execute a complete-split transaction using the transfer state of the current siTD. When the host controller commits to executing the complete-split transaction, it updates *QH.C-prog-mask* by bit-ORing with *cMicroFrameBit*. The transfer state is advanced based on the completion status of the complete-split transaction. To advance the transfer state of an IN siTD, the host controller must:

- Decrement the number of bytes received from *siTD.Total Bytes To Transfer*,
- Adjust *siTD.Current Offset* by the number of bytes received,

- Adjust *siTD.P* (page selector) field if the transfer caused the host controller to use the next page pointer, and
- Set any appropriate bits in the *siTD.Status* field, depending on the results of the transaction.

Note that if the host controller encounters a condition where *siTD.Total Bytes To Transfer* is zero, and it receives more data, the host controller must not write the additional data to memory. The *siTD.Status.Active* bit must be set to zero and the *siTD.Status.Babble Detected* bit must be set to a one. The fields *siTD.Total Bytes To Transfer*, *siTD.Current Offset*, and *siTD.P* (page selector) are not required to be updated as a result of this transaction attempt.

The host controller must accept (assuming good data packet CRC and sufficient room in the buffer as indicated by the value of *siTD.Total Bytes To Transfer*) MDATA and DATA0/1 data payloads up to and including 192 bytes. A host controller implementation may optionally set *siTD.Status Active* to a zero and *siTD.Status.Babble Detected* to a one when it receives MDATA or DATA0/1 with a data payload of more than 192 bytes. The following responses have the noted effects:

- ERR. The full-speed transaction completed with a time-out or bad CRC and this is a reflection of that error to the host. The host controller sets the *ERR* bit in the *siTD.Status* field and sets the *Active* bit to a zero.
- Transaction Error (XactErr). The complete-split transaction encounters a Timeout, CRC16 failure, etc. The *siTD.Status* field *XactErr* field is set to a one and the complete-split transaction must be retried immediately. The host controller must use an internal error counter to count the number of retries as a counter field is not provided in the siTD data structure. The host controller will not retry more than two times. If the host controller exhausts the retries or the end of the micro-frame occurs, the *Active* bit is set to zero.
- DATAx (0 or 1). This response signals that the final data for the split transaction has arrived. The transfer state of the siTD is advanced and the *Active* bit is set to a zero. If the *Bytes To Transfer* field has not decremented to zero (including the reception of the data payload in the DATAx response), then less data than was expected, or allowed for was actually received. This *short packet* event does not set the USBINT status bit in the USBSTS register to a one. The host controller will not detect this condition.
- NYET (and Last). On each NYET response, the host controller also checks to determine whether this is the last complete-split for this split transaction. Last was defined in Section [Periodic Isochronous - Do Complete Split](#) . If it is the last complete-split (with a NYET response), then the transfer state of the siTD is not advanced (never received any data) and the *Active* bit is set to a zero. No bits are set in the *Status* field because this is essentially a skipped transaction. The transaction translator must have responded to all the scheduled complete-splits with NYETs,

meaning that the start-split issued by the host controller was not received. This result should be interpreted by system software as if the transaction was completely skipped. The test for whether this is the last complete split can be performed by XORing C-mask with C-prog-mask. A zero result indicates that all complete-splits have been executed.

- MDATA (and Last). See above description for testing for Last. This can only occur when there is an error condition. Either there has been a babble condition on the full-speed link, which delayed the completion of the full-speed transaction, or software set up the *S-mask* and/or *C-masks* incorrectly. The host controller must set *XactErr* bit to a one and the *Active* bit is set to a zero.
- NYET (and not Last). See above description for testing for Last. The complete-split transaction received a NYET response from the transaction translator. Do not update any transfer state (except for *C-prog-mask*) and stay in this state.
- MDATA (and not Last). The transaction translator responds with an MDATA when it has partial data for the split transaction. For example, the full-speed transaction data payload spans from micro-frame X to X+1 and during micro-frame X, the transaction translator will respond with an MDATA and the data accumulated up to the end of micro-frame X. The host controller advances the transfer state to reflect the number of bytes received.

If Test A succeeds, but Test B fails, it means that one or more of the complete-splits have been skipped. The host controller sets the *Missed Micro-Frame* status bit and sets the *Active* bit to a zero.

#### 52.4.3.12.3.6 Complete-Split for Scheduling Boundary Cases 2a, 2b

Boundary cases 2a and 2b (INs only) (see [Figure 52-25](#)) require that the host controller use the transaction state context of the previous siTD to finish the split transaction. The table below enumerates the transaction state fields.

**Table 52-50. Summary siTD Split Transaction State**

Buffer State	Status	Execution Progress
Total Bytes To Transfer	All bits in the status field	C-prog-mask
P (page select)		
Current Offset		
TP (transaction position)		
T-count (transaction count)		

#### NOTE

*TP* and *T-count* are used only for Host to Device (OUT) endpoints.



If software has budgeted the schedule of this data stream with a frame wrap case, then it must initialize the *siTD.Back Pointer* field to reference a valid siTD and will have the *siTD.Back Pointer.T-bit* in the *siTD.Back Pointer*

field set to a zero. Otherwise, software must set the *siTD.Back Pointer.T-bit* in the *siTD.Back Pointer* field to a one. The host controller's rules for interpreting when to use the *siTD.Back Pointer* field are listed below. These rules apply only when the siTD's *Active* bit is a one and the *SplitXState* is Do Complete Split.

- When *cMicroFrameBit* is a 1h and the *siTDX.Back Pointer.T-bit* is a zero, or
- If *cMicroFrameBit* is a 2h and *siTDX.S-mask[0]* is a zero

When either of these conditions apply, then the host controller must use the transaction state from *siTD<sub>X-1</sub>*.

In order to access *siTD<sub>X-1</sub>*, the host controller reads on-chip the siTD referenced from *siTD<sub>X</sub>.Back Pointer*.

The host controller must save the entire state from *siTD<sub>X</sub>* while processing *siTD<sub>X-1</sub>*. This is to accommodate for case 2b processing. The host controller must not recursively walk the list of *siTD.Back Pointers*.

If *siTD<sub>X-1</sub>* is active (*Active* bit is a one and *SplitXStat* is Do Complete Split), then both Test A and Test B are applied as described above. If these criteria to execute a complete-split are met, the host controller executes the complete split and evaluates the results as described above. The transaction state (see [Table 52-50](#)) of *siTD<sub>X-1</sub>* is appropriately advanced based on the results and written back to memory. If the resultant state of *siTD<sub>X-1</sub>*'s *Active* bit is a one, then the host controller returns to the context of *siTD<sub>X</sub>*, and follows its next pointer to the next schedule item. No updates to *siTD<sub>X</sub>* are necessary.

If *siTD<sub>X-1</sub>* is active (*Active* bit is a one and *SplitXStat* is Do Start Split), then the host controller must set *Active* bit to a zero and *Missed Micro-Frame* status bit to a one and the resultant status written back to memory.

If *siTD<sub>X-1</sub>*'s *Active* bit is a zero, (because it was zero when the host controller first visited *siTD<sub>X-1</sub>* via *siTD<sub>X</sub>*'s back pointer, it transitioned to zero as a result of a detected error, or the results of *siTD<sub>X-1</sub>*'s complete-split transaction transitioned it to zero), then the host controller returns to the context of *siTD<sub>X</sub>* and transitions its *SplitXState* to Do Start Split. The host controller then determines whether the case 2b start split boundary condition exists (that is, if *cMicroframeBit* is a 1b and *siTD<sub>X</sub>.S-mask[0]* is a 1b). If this criterion is met the host controller immediately executes a start-split transaction and appropriately advances the transaction state of *siTD<sub>X</sub>*, then follows *siTD<sub>X</sub>.Next Pointer* to the next schedule item. If the criterion is not met, the host controller simply follows *siTD<sub>X</sub>.Next Pointer* to the next schedule item. Note that in the case of a 2b boundary case, the split-transaction of *siTD<sub>X-1</sub>* will have its *Active* bit set to zero when the host controller returns

to the context of  $siTD_X$ . Also, note that software should not initialize an siTD with *C-mask* bits 0 and 1 set to a one and an *S-mask* with bit zero set to a one. This scheduling combination is not supported and the behavior of the host controller is undefined.

### 52.4.3.12.3.7 Split Transaction for Isochronous - Processing Examples

There is an important difference between how the hardware/software manages the isochronous split transaction state machine and how it manages the asynchronous and interrupt split transaction state machines.

The asynchronous and interrupt split transaction state machines are encapsulated within a single queue head. The progress of the data stream depends on the progress of each split transaction. In some respects, the split-transaction state machine is sequenced via the Execute Transaction queue head traversal state machine (see [Figure 52-18](#)).

Isochronous is a pure time-oriented transaction/data stream. The interface data structures are optimized to efficiently describe transactions that need to occur at specific times. The isochronous split-transaction state machine must be managed across these time-oriented data structures. This means that system software must correctly describe the scheduling of split-transactions across more than one data structure.

Then the host controller must make the appropriate state transitions at the appropriate times, in the correct data structures.

For example, the table below illustrates a couple of frames worth of scheduling required to schedule a case 2a full-speed isochronous data stream.

**Table 52-51. Example Case 2a - Software Scheduling siTDs for an IN Endpoint**

siTDX		Micro-Frames								Initial
#	Masks	0	1	2	3	4	5	6	7	SplitXState
X	S-Mask	-	-	-	-	1	-	-	-	Do Start Split
	C-Mask	1	1	-	-	-	-	1	1	
X+1	S-Mask	-	-	-	-	1	-	-	-	Do Complete Split
	C-Mask	1	1					1	1	
X+2	S-Mask	-	-	-	-	1	-	-	-	Do Complete Split
	C-Mask	1	1					1	1	
X+3	S-Mask	Repeats previous pattern								Do Complete Split
	C-Mask									

This example shows the first three siTDs for the transaction stream. Because this is the case-2a frame-wrap case, *S-masks* of all siTDs for this endpoint have a value of 10h (a one bit in micro-frame 4) and *C-mask* value of C3h (one-bits in micro-frames 0,1, 6 and 7). Additionally, software ensures that the *Back Pointer* field of each siTD references the appropriate siTD data structure (and the *Back PointerT-bits* are set to zero).

The initial *SplitXState* of the first siTD is Do Start Split. The host controller will visit the first siTD eight times during frame X. The C-mask bits in micro-frames 0 and 1 are ignored because the state is Do Start Split. During micro-frame 4, the host controller determines that it can run a start-split (and does) and changes *SplitXState* to Do Complete Split. During micro-frames 6 and 7, the host controller executes complete-splits. Notice the siTD for frame X+1 has its *SplitXState* initialized to Do Complete Split. As the host controller continues to traverse the schedule during *H-Frame* X+1, it will visit the second siTD eight times. During micro-frames 0 and 1 it will detect that it must execute complete-splits.

During *H-Frame* X+1, micro-frame 0, the host controller detects that siTD<sub>X+1</sub>'s *Back Pointer.T-bit* is a zero, saves the state of siTD<sub>X+1</sub> and fetches siTD<sub>X</sub>. It executes the complete split transaction using the transaction state of siTD<sub>X</sub>. If the siTD<sub>X</sub> split transaction is complete, siTD's *Active* bit is set to zero and results written back to siTD<sub>X</sub>. The host controller retains the fact that siTD<sub>X</sub> is retired and transitions the *SplitXState* in the siTD<sub>X+1</sub> to Do Start Split. At this point, the host controller is prepared to execute the start-split for siTD<sub>X+1</sub> when it reaches micro-frame 4. If the split-transaction completes early (transaction-complete is defined in Section [Periodic Isochronous - Do Complete Split](#)), that is, before all the scheduled complete-splits have been executed, the host controller will transition siTD<sub>X</sub>.*SplitXState* to Do Start Split early and naturally skip the remaining scheduled complete-split transactions. For this example, siTD<sub>X+1</sub> does not receive a DATA0 response until *H-Frame* X+2, micro-frame 1.

During *H-Frame* X+2, micro-frame 0, the host controller detects that siTD<sub>X+2</sub>'s *Back Pointer.T-bit* is a zero, saves the state of siTD<sub>X+2</sub> and fetches siTD<sub>X+1</sub>. As described above, it executes another split transaction, receives an MDATA response, updates the transfer state, but does not modify the *Active* bit. The host controller returns to the context of siTD<sub>X+2</sub>, and traverses its next pointer without any state change updates to siTD<sub>X+2</sub>. S

During *H-Frame* X+2, micro-frame 1, the host controller detects siTD<sub>X+2</sub>'s *S-mask[0]* is a zero, saves the state of siTD<sub>X+2</sub> and fetches siTD<sub>X+1</sub>. It executes another complete-split transaction, receives a DATA0 response, updates the transfer state and sets the *Active* bit to a zero. It returns to the state of siTD<sub>X+2</sub> and changes its *SplitXState* to Do Start Split. At this point, the host controller is prepared to execute start-splits for siTD<sub>X+2</sub> when it



reaches micro-frame 4. <TBD... describe how software detects that there was missing micro-frames (don't think we care about missing out micro-frames. There is enough residual state to identify than not all transactions were executed.).

### 52.4.3.13 Host Controller Pause

When the host controller's *HCHalted* bit in the USBSTS register is a zero, the host controller is sending SOF (Start OF Frame) packets down all enabled ports.

When the schedules are enabled, the EHCI host controller will access the schedules in main memory each micro-frame. This constant pinging of main memory is known to create ARM platform power management problems for mobile systems. Specifically, mobile systems aggressively manage the state of the ARM platform, based on recent history usage. In the more aggressive power saving modes, the ARM platform can disable its caches. Current PC architectures assume that bus-master accesses to main memory must be cache-coherent. So, when bus masters are busy touching memory, the ARM platform power management software can detect this activity over time and inhibit the transition of the ARM platform into its lowest power savings mode. USB controllers are bus-masters and the frequency at which they access their memory-based schedules keeps the ARM platform power management software from placing the ARM platform into its lowest power savings state.

USB Host controllers don't access main memory when they are suspended. However, there are a variety of reasons why placing the USB controllers into suspend won't work, but they are beyond the scope of this document. The base requirement is that the USB controller needs to be kept out of main memory, while at the same time, the USB bus is kept from going into suspend.

EHCI controllers provide a large-grained mechanism that can be manipulated by system software to change the memory access pattern of the host controller. System software can manipulate the schedule enable bits in the USBCMD register to turn on/off the scheduling traversal. A software heuristic can be applied to implement an on/off duty cycle that allows the USB to make reasonable progress and allow the ARM platform power management to get the ARM platform into its lowest power state. This method is not intended to be applied at all times to throttle USB, but should only be applied in very specific configurations and usage loads. For example, when only a keyboard or mouse is attached to the USB, the heuristic could detect times when the USB is attempting to move data only very infrequently and can adjust the duty cycle to allow the ARM platform to reach its low power state for longer periods of time. Similarly, it could detect increases in the USB load and adjust the duty cycle appropriately, even to the point where the schedules are never disabled. The assumption here is that the USB is moving data and the ARM platform will be required to process the data streams.

It is suggested that in order to provide a complete solution for the system, the companion host controllers should also provide a similar method to allow system software to inhibit the companion host controller from accessing its shared memory based data structures (schedule lists or otherwise).

#### 52.4.3.14 Port Test Modes -Host Operational Model

EHCI host controllers must implement the port test modes Test\_J\_State, Test\_K\_State, Test\_Packet, Test\_Force\_Enable, and Test\_SE0\_NAK as described in the USB Specification Revision 2.0.

The system is only allowed to test ports that are owned by the EHCI controller (for example, *CF-bit* is a one and *PortOwner* bit is a zero). System software is allowed to have at most one port in test mode at a time. Placing more than one port in test mode will yield undefined results. The required, per port test sequence is (assuming the *CF-bit* in the USB\_n\_CONFIGFLAG register is a one):

- Disable the periodic and asynchronous schedules by setting the *Asynchronous Schedule Enable* and *Periodic Schedule Enable* bits in the USBCMD register to a zero.
- Place all enabled root ports into the suspended state by setting the *Suspend* bit in each appropriate USB\_n\_PORTSC register to a one.
- Set the *Run/Stop* bit in the USBCMD register to a zero and wait for the *HCHalted* bit in the USBSTS register, to transition to a one. Note that an EHCI host controller implementation may optionally allow port testing with the *Run/Stop* bit set to a one. However, all host controllers must support port testing with *Run/Stop* set to a zero and *HCHalted* set to a one.
- Set the *Port Test Control* field in the port under test PORTSC register to the value corresponding to the desired test mode. If the selected test is Test\_Force\_Enable, then the *Run/Stop* bit in the USBCMD register must then be transitioned back to one, in order to enable transmission of SOFs out of the port under test.
- When the test is complete, system software must ensure the host controller is halted (*HCHalted* bit is a one) then it terminates and exits test mode by setting *HCRreset* to a one.

#### 52.4.3.15 Interrupts-Host Operational Model

The EHCI Host Controller hardware provides interrupt capability based on a number of sources.

There are several general groups of interrupt sources:

- Interrupts as a result of executing transactions from the schedule (success and error conditions),
- Host controller events (Port change events, etc.), and
- Host Controller error events

All transaction-based sources are maskable through the Host Controller's Interrupt Enable register (USBINTR, see Section [Interrupt Enable Register \(USBC\\_n\\_USBINTR\)](#)). Additionally, individual transfer descriptors can be marked to generate an interrupt on completion. This section describes each interrupt source and the processing that occurs in response to the interrupt.

During normal operation, interrupts may be immediate or deferred until the next interrupt threshold occurs. The interrupt threshold is a tunable parameter via the *Interrupt Threshold Control* field in the USBCMD register. The value of this register controls when the host controller will generate an interrupt on behalf of normal transaction execution. When a transaction completes during an interrupt interval period, the interrupt signaling the completion of the transfer will not occur until the interrupt threshold occurs. For example, the default value is eight micro-frames. This means that the host controller will not generate interrupts any more frequently than once every eight micro-frames.

Section [Host System Error](#) details effects of a host system error.

If an interrupt has been scheduled to be generated for the current interrupt threshold interval, the interrupt is not signaled until after the status for the last complete transaction in the interval has been written back to host memory. This may sometimes result in the interrupt not being signaled until the next interrupt threshold.

Initial interrupt processing is the same, regardless of the reason for the interrupt. When an interrupt is signaled by the hardware, ARM platform control is transferred to host controller's USB interrupt handler. The precise mechanism to accomplish the transfer is OS specific. For this discussion it is just assumed that control is received. When the interrupt handler receives control, its first action is to read the USBSTS (USB Status Register). It then acknowledges the interrupt by clearing all of the interrupt status bits by writing ones to these bit positions. The handler then determines whether the interrupt is due to schedule processing or some other event. After acknowledging the interrupt, the handler (via an OS-specific mechanism), schedules a deferred procedure call (DPC) which will execute later. The DPC routine processes the results of the schedule execution. The precise mechanisms used are beyond the scope of this document.

Note: the host controller is not required to de-assert a currently active interrupt condition when software sets the interrupt enables (in the USBINTR register, see Section [Interrupt Enable Register \(USBC\\_n\\_USBINTR\)](#)) to a zero. The only reliable method software

should use for acknowledging an interrupt is by transitioning the appropriate status bits in the USBSTS register (Section [USB Status Register \(USBC\\_n\\_USBSTS\)](#)) from a one to a zero.

### 52.4.3.15.1 Transfer/Transaction Based Interrupts

These interrupt sources are associated with transfer and transaction progress. They are all dependent on the next interrupt threshold.

#### 52.4.3.15.1.1 Transaction Error

A transaction error is any error that caused the host controller to think that the transfer did not complete successfully.

The table below lists the events/responses that the host can observe as a result of a transaction. The effects of the error counter and interrupt status are summarized in the following paragraphs. Most of these errors set the *XactErr* status bit in the appropriate interface data structure.

There is a small set of protocol errors that relate only when executing a queue head and fit under the umbrella of a WRONG PID error that are significant to explicitly identify. When these errors occur, the *XactErr* status bit in the queue head is set and the *CErr* field is decremented. When the *PIDCode* indicates a SETUP, the following responses are protocol errors and result in *XactErr* bit being set to a one and the *CErr* field being decremented.

- *EPS* field indicates a high-speed device and it returns a Nak handshake to a SETUP.
- *EPS* field indicates a high-speed device and it returns a Nyet handshake to a SETUP.
- *EPS* field indicates a low- or full-speed device and the complete-split receives a Nak handshake.

**Table 52-52. Summary of Transaction Errors**

Event / Result	Queue Head/qTD/iTD/siTD Side-effects		USB Status Register (USBSTS)
	Cerr	Status Field	USBERRINT
CRC	-1	XactErr set to a one.	1 <sup>1</sup>
Timeout	-1	XactErr set to a one.	1 <sup>1</sup>
Bad PID <sup>2</sup>	-1	XactErr set to a one.	1 <sup>1</sup>
Babble	N/A	Section <a href="#">Serial Bus Babble</a>	1
Buffer Error	N/A	Section <a href="#">Data Buffer Error</a>	

1. If occurs in a queue head, then *USBERRINT* is asserted only when *CErr* counts down from a one to a zero. In addition the queue is halted, see [Halting a Queue Head](#).
2. The host controller received a response from the device, but it could not recognize the PID as a valid PID.

### 52.4.3.15.1.2 Serial Bus Babble

When a device transmits more data on the USB than the host controller is expecting for this transaction, it is defined to be babbling. In general, this is called a *Packet Babble*.

When a device sends more data than the *Maximum Length* number of bytes, the host controller sets the *Babble Detected* bit to a one and halts the endpoint if it is using a queue head (see [Halting a Queue Head](#)). *Maximum Length* is defined as the minimum of *Total Bytes to Transfer* and *Maximum Packet Size*. The *CErr* field is not decremented for a packet babble condition (only applies to queue heads). A babble condition also exists if IN transaction is in progress at High-speed EOF2 point. This is called a frame babble. A frame babble condition is recorded into the appropriate schedule data structure. In addition, the host controller must disable the port to which the frame babble is detected.

The *USBERRINT* bit in the USB\_n\_USBSTS register is set to a one and if the *USB Error Interrupt Enable* bit in the USB\_n\_USBINTR register is a one, then a hardware interrupt is signaled to the system at the next interrupt threshold. The host controller must never start an OUT transaction that will babble across a micro-frame EOF.

#### NOTE

When a host controller detects a data PID mismatch, it must either: disable the packet babble checking for the duration of the bus transaction or do packet babble checking based solely on *Maximum Packet Size*. The USB core specification defines the requirements on a data receiver when it receives a data PID mismatch (for example, expects a DATA0 and gets a DATA1 or visa-versa). In summary, it must ignore the received data and respond with an ACK handshake, in order to advance the transmitter's data sequence.

The EHCI interface allows System software to provide buffers for a Control, Bulk or Interrupt IN endpoint that are not an even multiple of the maximum packet size specified by the device. Whenever a device misses an ACK for an IN endpoint, the host and device are out of synchronization with respect to the progress of the data transfer. The host controller may have advanced the transfer to a buffer that is less than maximum packet size. The device will re-send its maximum packet size data packet, with the original data PID, in response to the next IN token. In order to properly manage the bus protocol, the host controller must disable the packet babble check when it observes the data PID mismatch.

### 52.4.3.15.1.3 Data Buffer Error

This event indicates that an overrun of incoming data or a underrun of outgoing data has occurred for this transaction.

This would generally be caused by the host controller not being able to access required data buffers in memory within necessary latency requirements. These conditions are not considered transaction errors, and do not effect the error count in the queue head. When these errors do occur, the host controller records the fact the error occurred by setting the *Data Buffer Error* bit in the queue head, iTD or siTD.

If the data buffer error occurs on a non-isochronous IN, the host controller will not issue a handshake to the endpoint. This will force the endpoint to resend the same data (and data toggle) in response to the next IN to the endpoint.

If the data buffer error occurs on an OUT, the host controller must corrupt the end of the packet so that it cannot be interpreted by the device as a good data packet. Simply truncating the packet is not considered acceptable. An acceptable implementation option is to 1's complement the CRC bytes and send them. There are other options suggested in the Transaction Translator section of the USB Specification Revision 2.0.

### 52.4.3.15.1.4 USB Interrupt (Interrupt on Completion (IOC))

Transfer Descriptors (iTDs, siTDs, and queue heads (qTDs)) contain a bit that can be set to cause an interrupt on their completion. The completion of the transfer associated with that schedule item causes the USB Interrupt (USBINT) bit in the USB\_n\_USBSTS register to be set to a one.

In addition, if a short packet is encountered on an IN transaction associated with a queue head, then this event also causes USBINT to be set to a one. If the USB Interrupt Enable bit in the USB\_n\_USBINTR register is set to a one, a hardware interrupt is signaled to the system at the next interrupt threshold. If the completion is because of errors, the *USBERRINT* bit in the USB\_n\_USBSTS register is also set to a one.

### 52.4.3.15.1.5 Short Packet

Reception of a data packet that is less than the endpoint's Max Packet size during Control, Bulk or Interrupt transfers signals the completion of the transfer. Whenever a short packet completion occurs during a queue head execution, the *USBINT* bit in the USB\_n\_USBSTS register is set to a one.

If the *USB Interrupt Enable* bit is set in the USB\_n\_USBINTR register, a hardware interrupt is signaled to the system at the next interrupt threshold.

### 52.4.3.15.2 Host Controller Event Interrupts

These interrupt sources are independent of the interrupt threshold (with the one exception being the Interrupt on Async Advance, see Section [Interrupt on Async Advance](#) ).

#### 52.4.3.15.2.1 Port Change Events

Port registers contain status and status change bits. When the status change bits are set to a one, the host controller sets the *Port Change Detect* bit in the USBSTS register to a one.

If the *Port Change Interrupt Enable* bit in the USB\_n\_USBINTR register is a one, then the host controller will issue a hardware interrupt. The port status change bits include:

- Connect Status Change
- Port Enable/Disable Change
- Over-current Change
- Force Port Resume

#### 52.4.3.15.2.2 Frame List Rollover

This event indicates that the host controller has wrapped the frame list. The current programmed size of the frame list effects how often this interrupt occurs.

If the frame list size is 1024, then the interrupt will occur every 1024 milliseconds, if it is 512, then it will occur every 512 milliseconds, etc. When a frame list rollover is detected, the host controller sets the *Frame List Rollover* bit in the USB.USBSTS register to a one. If the *Frame List Rollover Enable* bit in the USB.USBINTR register is set to a one, the host controller issues a hardware interrupt. This interrupt is not delayed to the next interrupt threshold.

#### 52.4.3.15.2.3 Interrupt on Async Advance

This event is used for deterministic removal of queue heads from the asynchronous schedule. Whenever the host controller advances the on-chip context of the asynchronous schedule, it evaluates the value of the *Interrupt on Async Advance Doorbell* bit in the USB.USBCMD register.

If it is a one, it sets the *Interrupt on Async Advance* bit in the USB.USBSTS register to a one. If the *Interrupt on Async Advance Enable* bit in the USB.USBINTR register is a one, the host controller issues a hardware interrupt at the next interrupt threshold. A detailed explanation of this feature is described in Section [Removing Queue Heads from Asynchronous Schedule](#) .

### 52.4.3.15.2.4 Host System Error

The host controller is a bus master and any interaction between the host controller and the system may experience errors.

The type of host error may be catastrophic to the host controller (such as a Master Abort) making it impossible for the host controller to continue in a coherent fashion. In the presence of non-catastrophic host errors, such as parity errors, the host controller could potentially continue operation. The recommended behavior for these types of errors is to escalate it to a catastrophic error and halt the host controller. Host-based error must result in the following actions:

- The *Run/Stop* bit in the USB.USBCMD register is set to a zero.
- The following bits in the USB.USBSTS register are set:
  - *Host System Error* bit is to a one.
  - *HCHalted* bit is set to a one.
- If the *Host System Error Enable* bit in the USB.USBINTR register is a one, then the host controller will issue a hardware interrupt. This interrupt is not delayed to the next interrupt threshold. The following table summarizes the required actions taken on the various host errors.

**Table 52-53. Summary Behavior of EHCI Host Controller on Host System Errors**

Cycle Type	Master Abort	Target Abort	Data Phase Parity
Frame list pointer fetch (read)	Fatal	Fatal	Fatal [o]
siTD fetch (read)	Fatal	Fatal	Fatal [o]
siTD status write-back (write)	Fatal [o]	Fatal [o]	Fatal [o]
iTD fetch (read)	Fatal	Fatal	Fatal [o]
iTD status write-back (write)	Fatal [o]	Fatal [o]	Fatal [o]
qTD fetch (read)	Fatal	Fatal	Fatal [o]
qHD status write-back (write)	Fatal [o]	Fatal [o]	Fatal [o]
Data write	Fatal [o]	Fatal [o]	Fatal [o]
Data read	Fatal	Fatal	Fatal [o]

Potentially, a host controller implementation could continue operation without a halt. However, the recommended behavior is to halt the host controller.

#### NOTE

After a *Host System Error*, Software must reset the host controller through *HCRreset* in the USB.USBCMD register before re-initializing and restarting the host controller.



## 52.4.4 EHCI Deviation

For the purposes a dual-role Host/Device controller with support for On-The-Go applications, it is necessary to deviate from the EHCI specification Enhanced Host Controller Interface Specification for Universal Serial Bus, Revision 0.95, November 2000, Intel Corporation. <http://www.intel.com>. Device operation & On-The-Go operation is not specified in the EHCI and thus the implementation supported in this core is proprietary.

The host mode operation of the core is near EHCI compatible with few minor differences documented in this section.

The particulars of the deviations occur in the areas summarized here:

- Embedded Transaction Translator - Allows direct attachment of FS and LS devices in host mode without the need for a companion controller.
- Device operation - In host mode the device operational registers are generally disabled and thus device mode is mostly transparent when in host mode. However, there are a couple exceptions documented in the following sections.
- Embedded design interface - This core does not have a PCI Interface and therefore the PCI configuration registers described in the EHCI specification are not applicable.
- On-The-Go Operation - This design includes an On-The-Go controller for Port #1.

### 52.4.4.1 Embedded Transaction Translator Function

The USB-HS OTG High-Speed USB On-The-Go OTG controller supports directly connected full and low speed devices without requiring a companion controller by including the capabilities of a USB 2.0 high speed hub transaction translator.

Although there is no separate Transaction Translator block in the system, the transaction translator function normally associated with a high speed hub has been implemented within the DMA and Protocol engine blocks. The embedded transaction translator function is an extension to EHCI interface, but makes use of the standard data structures and operational models that exist in the EHCI specification to support full and low speed devices.

#### 52.4.4.1.1 Capability Registers

The following additions have been added to the capability registers to support the embedded Transaction Translator Function:

- N\_TT added to USB.HCSPARAMS - Host Control Structural Parameters
- N\_PTT added to USB.HCSPARAMS - Host Control Structural Parameters

### 52.4.4.1.2 Operational Registers

The following additions have been added to the operational registers to support the embedded TT:

- Addition of two-bit Port Speed (PSPD) to the [Port Status & Control \(USBC\\_n\\_PORTSC1\)](#) register.

### 52.4.4.1.3 Discovery-EHCI Deviation

In a standard EHCI controller design, the EHCI host controller driver detects a Full speed (FS) or Low speed (LS) device by noting if the port enable bit is set after the port reset operation.

The port enable will only be set in a standard EHCI controller implementation after the port reset operation and when the host and device negotiate a High-Speed connection (that is, Chirp completes successfully).

Because this controller has an embedded Transaction Translator, the port enable will always be set after the port reset operation regardless of the result of the host device chirp result and the resulting port speed will be indicated by the PSPD field in USB.PORTSCx.

Therefore, the standard EHCI host controller driver requires an alteration to handle directly connected Full and Low speed devices or hubs.

The change is a fundamental one in that is summarized in the following table.

**Table 52-54. Summary of EHCI**

Standard EHCI	EHCI with embedded Transaction Translator
After port enable bit is set following a connection and reset sequence, the device/hub is assumed to be HS.	After port enable bit is set following a connection and reset sequence, the device/hub speed is noted from USB.PORTSCx.
FS and LS devices are assumed to be downstream from a HS hub thus, all port-level control is performed through the Hub Class to the nearest Hub.	FS and LS device can be either downstream from a HS hub or directly attached. When the FS/LS device is downstream from a HS hub, then port-level control is done using the Hub Class through the nearest Hub. When a FS/LS device is directly attached, then port-level control is accomplished using USB.PORTSCx.
FS and LS devices are assumed to be downstream from a HS hub with HubAddr=X. [where HubAddr > 0 and HubAddr is the address of the Hub where the bus transitions from HS to FS/LS (ie. Split target hub)]	FS and LS device can be either downstream from a HS hub with HubAddr = X [HubAddr > 0] or directly attached [where HubAddr = 0 and HubAddr is the address of the Root Hub where the bus transitions from HS to FS/LS (ie. Split target hub is the root hub) ]

#### 52.4.4.1.4 Data Structures

The same data structures used for FS/LS transactions through a HS hub are also used for transactions through the Root Hub with sm embedded Transaction Translator.

Here it is demonstrated how the Hub Address and Endpoint Speed fields should be set for directly attached FS/LS devices and hubs:

1. QH (for direct attach FS/LS) - Async. (Bulk/Control Endpoints) Periodic (Interrupt)
  - Hub Address = 0
  - Transactions to direct attached device/hub.
    - QH.EPS = Port Speed
  - Transactions to a device downstream from direct attached FS hub.
    - QH.EPS = Downstream Device Speed

#### NOTE

When QH.EPS = 01 (LS) and PORTSCx.PSPD = 00 (FS), a LS-pre-pid will be sent before the transmitting LS traffic.

Maximum Packet Size must be less than or equal 64 or undefined behaviour may result.

2. siTD (for direct attach FS) - Periodic (ISO Endpoint)
  - All FS ISO transactions:
    - Hub Address = 0
    - siTD.EPS = 00 (full speed)
      - Maximum Packet Size must less than or equal to 1023 or undefined behaviour may result.

#### 52.4.4.1.5 Operational Model

The operational models are well defined for the behavior of the Transaction Translator (see USB 2.0 specification Universal Serial Bus Specification, Revision 2.0, April 2000, Compaq, Hewlett-Packard, Intel, Lucent, Microsoft, NEC, Philips. <http://www.usb.org>) and for the EHCI controller moving packets between system memory and a USB-HS hub. Because the embedded Transaction Translator exists within the host controller there is no physical bus between EHCI host controller driver and the USB FS/LS bus. These sections will briefly discuss the operational model for how the EHCI and Transaction Translator operational models are combined without the physical bus between. The following sections assume the reader is familiar with both the EHCI and USB 2.0 Transaction Translator operational models.

#### 52.4.4.1.5.1 Micro-frame Pipeline

The EHCI operational model uses the concept of H-frames and B-frames to describe the pipeline between the Host (H) and the Bus (B). The embedded Transaction Translator shall use the same pipeline algorithms specified in the USB 2.0 specification for a Hub-based Transaction Translator.

All periodic transfers always begin at B-frame 0 (after SOF) and continue until the stored periodic transfers are complete. As an example of the micro-frame pipeline implemented in the embedded Transaction Translator, all periodic transfers that are tagged in EHCI to execute in H-frame 0 will be ready to execute on the bus in B-frame 0.

It is important to note that when programming the S-mask and C-masks in the EHCI data structures to schedule periodic transfers for the embedded Transaction Translator, the EHCI host controller driver must follow the same rules specified in EHCI for programming the S-mask and C-mask for downstream Hub-based Transaction Translators.

Once periodic transfers are exhausted, any stored asynchronous transfer will be moved. Asynchronous transfers are opportunistic in that they shall execute whenever possible and their operation is not tied to H-frame and B-frame boundaries with the exception that an asynchronous transfer can not babble through the SOF (start of B-frame 0.)

#### 52.4.4.1.5.2 Split State Machines

The start and complete split operational model differs from EHCI slightly because there is no bus medium between the EHCI controller and the embedded Transaction Translator.

Where a start or complete-split operation would occur by requesting the split to the HS hub, the start/complete split operation is simple an internal operation to the embedded Transaction Translator. The following table summarizes the conditions where handshakes are emulated from internal state instead of actual handshakes to HS split bus traffic.

**Table 52-55. Summary of the Conditions of Handshakes<sup>1</sup>**

Condition	Emulate TT Response
Start-Split: All asynchronous buffers full.	NAK
Start-Split: All periodic buffers full.	ERR
Start-Split: Success for start of Async. Transaction.	ACK
Start-Split: Start Periodic Transaction.	No Handshake (Ok)
Complete-Split: Failed to find transaction in queue.	Bus Time Out
Complete-Split: Transaction in Queue is Busy.	NYET
Complete-Split: Transaction in Queue is Complete.	[Actual Handshake from LS/FS device]

1. The un-shaded cells represent Start-Splits and the shaded cells represent Complete-Splits

#### **52.4.4.1.5.3 Asynchronous Transaction Scheduling and Buffer Management**

The following USB 2.0 specification items are implemented in the embedded Transaction Translator:

##### **52.4.4.1.5.4 USB 2.0 - 11.17.3**

- Sequencing is provided & a packet length estimator ensures no full-speed/low-speed packet babbles into SOF time.

##### **52.4.4.1.5.5 USB 2.0 - 11.17.4**

- Transaction tracking for 2 data pipes.

#### **52.4.4.1.5.6 Periodic Transaction Scheduling and Buffer Management**

The following USB 2.0 specification items are implemented in the embedded Transaction Translator:

##### **52.4.4.1.5.7 USB 2.0 - 11.18.6.[1-2]**

- Abort of pending start-splits
  - EOF (and not started in micro-frames 6)
  - Idle for more than 4 micro-frames
- Abort of pending complete-splits
  - EOF
  - Idle for more than 4 micro-frames

##### **52.4.4.1.5.8 USB 2.0 - 11.18.[7-8]**

- Transaction tracking for up to 16 data pipes.
- Complete-split transaction searching.

#### **NOTE**

There is no data schedule mechanism for these transactions other than the micro-frame pipeline. The embedded TT assumes the number of packets scheduled in a frame does not exceed the frame duration (1 ms) or else undefined behavior may result.

#### 52.4.4.1.5.9 Multiple Transaction Translators

The maximum number of embedded Transaction Translators that is currently supported is one as indicated by the N\_TT field in the [Host Controller Structural Parameters \(USBC\\_n\\_HCSPARAMS\)](#) register.

#### 52.4.4.2 Device Operation

The co-existence of a device operational controller within the host controller has little effect on EHCI compatibility for host operation except as noted in this section.

#### 52.4.4.3 USB\_USBMODE Register

Given that the dual-role controller is initialized in neither host nor device mode, the [USB Device Mode \(USBC\\_n\\_USBMODE\)](#) register must be programmed for host operation before the EHCI host controller driver can begin EHCI host operations.

##### 52.4.4.3.1 Non-Zero Fields the Register File

Some of the reserved fields and reserved addresses in the capability registers and operational register have use in device mode, the following must be adhered to:

- Write operations to all EHCI reserved fields (some of which are device fields) with the operation registers should always be written to zero. This is an EHCI requirement of the device controller driver that must be adhered to.
- Read operations by the host controller must properly mask EHCI reserved fields (some of which are device fields) because fields that are used exclusive for device are undefined in host mode .

##### 52.4.4.3.2 SOF Interrupt

This SOF Interrupt used for device mode is shared as a free running 125us interrupt for host mode.

EHCI does not specify this interrupt but it has been added for convenience and as a potential software time base. See [USB Status Register \(USBC\\_n\\_USBSTS\)](#) and [Interrupt Enable Register \(USBC\\_n\\_USBINTR\)](#) registers.

### 52.4.4.4 Embedded Design Interface

This is an Embedded USB Host Controller as defined by the EHCI specification and thus does not implement the PCI configuration registers.

#### 52.4.4.4.1 Frame Adjust Register

Given that the optional PCI configuration registers are not included in this implementation, there is no corresponding bit level timing adjustments like is provided by the Frame Adjust register in the PCI configuration registers. Starts of micro-frames are timed precisely to 125 us using the transceiver clock as a reference clock. That is, a 60 Mhz transceiver clock for 8-bit physical interfaces & full-speed serial interfaces or 30 Mhz transceiver clock for 16-bit physical interfaces.

### 52.4.4.5 Miscellaneous variations from EHCI

#### 52.4.4.5.1 Programmable Physical Interface Behaviour

This design supports multiple Physical interfaces which can operate in differing modes when the core is configured with software programmable Physical Interface Modes.

Software programmability allows the selection of the Physical interface part during the board design phase instead of during the chip design phase.

The control bits for selecting the Physical Interface operating mode have been added to the [Port Status & Control \(USBC\\_n\\_PORTSC1\)](#) register providing a capability that is not defined by EHCI.

#### 52.4.4.5.2 Discovery

##### 52.4.4.5.2.1 Port Reset

The port connect methods specified by EHCI require setting the port reset bit in the [Port Status & Control \(USBC\\_n\\_PORTSC1\)](#) register for a duration of 10ms. Due to the complexity required to support the attachment of devices that are not high speed there are counter already present in the design that can count the 10ms reset pulse to alleviate the requirement of the software to measure this duration. Therefore, the basic connection is then summarized as the following:

- [Port Change Interrupt] Port connect change occurs to notify the host controller driver that a device has attached.
- Software shall write a '1' to the reset the device.

- Software shall write a '0' to the reset the device after 10 ms.
  - This step, which is necessary in a standard EHCI design, may be omitted with this implementation. Should the EHCI host controller driver attempt to write a '0' to the reset bit while a reset is in progress, the write will simple be ignored and the reset will continue until completion.
- [Port Change Interrupt] Port enable change occurs to notify the host controller that the device in now operational and at this point the port speed has been determined.

#### 52.4.4.5.2.2 Port Speed Detection

After the port change interrupt indicates that a port is enabled, the EHCI stack should determine the port speed. Unlike the EHCI implementation, which will re-assign the port owner for any device that does not connect at High-Speed, this host controller supports direct attach of non High-Speed devices.

Therefore, the following differences are important regarding port speed detection:

- Port Owner is read-only and always reads 0.
- A 2-bit Port Speed indicator has been added to PORTSC to provide the current operating speed of the port to the host controller driver.
- A 1-bit High Speed indicator has been added to PORTSC to signify that the port is in High-Speed vs. Full/Low Speed - *This information is redundant with the 2-bit Port Speed indicator above.*

#### 52.4.4.5.3 Port Test Mode

Port Test Control mode behaves fully as described in EHCI. An alternate host controller driver procedure is not necessary or supported.

### 52.4.5 Device Data Structures

This section defines the interface data structures used to communicate control, status, and data between Device Controller Driver (DCD) Software and the Device Controller.

The data structure definitions in this chapter support a 32-bit memory buffer address space. The interface consists of device Queue Heads and Transfer Descriptors.

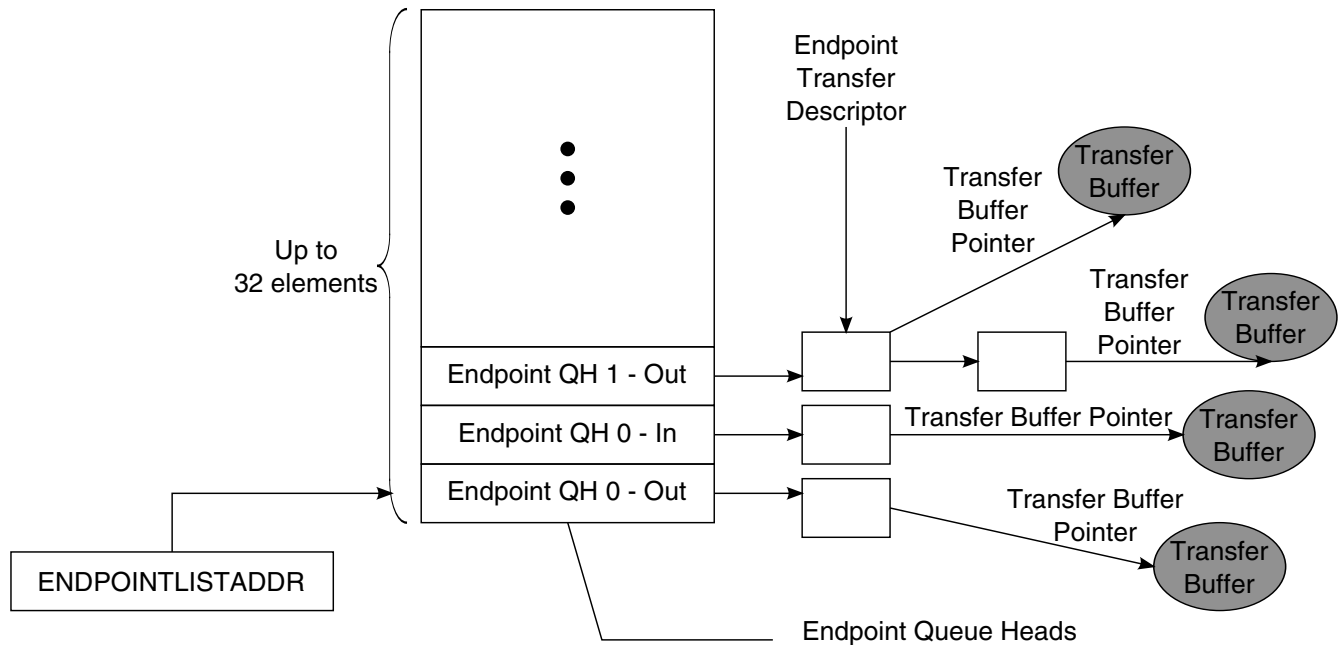
#### NOTE

Software must ensure that no interface data structure reachable by the Device Controller spans a 4K-page boundary.



The data structures defined in the chapter are (from the device controller's perspective) a mix of read-only and read/ writable fields. The device controller must preserve the read-only fields on all data structure writes.

The figure below shows the organization of the EndPoint Queue Head.



**Figure 52-28. EndPoint Queue Head Organization**

Endpoint queue heads are arranged in an array in a continuous area of memory pointed to by the USB.ENDPOINTLISTADDR pointer. The even -numbered device queue heads in the list support receive endpoints (OUT/SETUP) and the odd-numbered queue heads in the list are used for transmit endpoints (IN/INTERRUPT). The device controller will index into this array based upon the endpoint number received from the USB bus. All information necessary to respond to transactions for all primed transfers is contained in this list so the Device Controller can readily respond to incoming requests without having to traverse a linked list.

### NOTE

The Endpoint Queue Head List must be aligned to a 2k boundary.

#### 52.4.5.1 Endpoint Queue Head (dQH)

The device Endpoint Queue Head (dQH) is where all transfers for a given endpoint are managed. The dQH is a 48-byte data structure, but must be aligned on 64-byte boundaries.

During priming of an endpoint, the dTD (device transfer descriptor) is copied into the overlay area of the dQH, which starts at the nextTD pointer DWord and continues through the end of the buffer pointers DWords. After a transfer is complete, the dTD status DWord is updated in the dTD pointed to by the currentTD pointer. While a packet is in progress, the overlay area of the dQH is used as a staging area for the dTD so that the Device Controller can access needed information with little minimal latency.

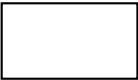
Table 52-56. Endpoint Queue Head (dQH)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Mult		zlt		0	Maximum Packet Length											ios	0															
Current dTD Pointer																												0				
Next dTD Pointer																												0		T <sup>1</sup>		
0	Total Bytes															ioc	0		MultO		0		Status									
Buffer Pointer (Page 0)																				Current Offset												
Buffer Pointer (Page 1)																				Reserved												
Buffer Pointer (Page 2)																				Reserved												
Buffer Pointer (Page 3)																				Reserved												
Buffer Pointer (Page 4) <sup>1</sup>																				Reserved												
Reserved																																
Set-up Buffer Bytes 3...0																																
Set-up Buffer Bytes 7...4																																

1. Transfer overlay starts at T and continues through Buffer Pointer (Page 4).



Host Controller Read/Write



Host Controller Read Only

52.4.5.1.1 Endpoint Capabilities/Characteristics

This DWord specifies static information about the endpoint, in other words, this information does not change over the lifetime of the endpoint. Device Controller software should not attempt to modify this information while the corresponding endpoint is enabled.

Table 52-57 describes the endpoint capabilities.

**Table 52-57. Endpoint Capabilities/Characteristics**

Bit	Description
31-30	Mult. This field is used to indicate the number of packets executed per transaction description as given by the following:  00 - Execute N Transactions as demonstrated by the USB variable length packet protocol where N is computed using the Maximum Packet Length (dQH) and the Total Bytes field (dTD)  01 Execute 1 Transaction. 10 Execute 2 Transactions. 11 Execute 3 Transactions.  <b>NOTE:</b> Non-ISO endpoints must set Mult="00". ISO endpoints must set Mult="01", "10", or "11" as needed.
29	Zero Length Termination Select. This bit is used to indicate when a zero length packet is used to terminate transfers where to total transfer length is a multiple . This bit is not relevant for Isochronous  0 - Enable zero length packet to terminate transfers equal to a multiple of the Maximum Packet Length. (default).  1 - Disable the zero length packet on transfers that are equal in length to a multiple Maximum Packet Length.
28-27	Reserved. These bit reserved for future use and should be set to zero.
26-16	Maximum Packet Length. This directly corresponds to the maximum packet size of the associated endpoint (wMaxPacketSize). The maximum value this field may contain is 0x400 (1024).
15	Interrupt On Setup (IOS). This bit is used on control type endpoints to indicate if USBINT is set in response to a setup being received.
14-0	Reserved. Bits reserved for future use and should be set to zero.

#### 52.4.5.1.2 Transfer Overlay-Endpoint Queue Head

The seven DWords in the overlay area represent a transaction working space for the device controller.

The general operational model is that the device controller can detect whether the overlay area contains a description of an active transfer. If it does not contain an active transfer, then it will not read the associated endpoint.

After an endpoint is readied, the dTD will be copied into this queue head overlay area by the device controller. Until a transfer is expired, software must not write the queue head overlay area or the associated transfer descriptor. When the transfer is complete, the device controller will write the results back to the original transfer descriptor and advance the queue.

See dTD for a description of the overlay fields.

#### 52.4.5.1.3 Current dTD Pointer

The current dTD pointer is used by the device controller to locate the transfer in progress. This word is for Device Controller (hardware) use only and should not be modified by DCD software.

The following table describes the dTD Pointer.

**Table 52-58. Next dTD Pointer**

Bit	Description
31-5	Current dTD. This field is a pointer to the dTD that is represented in the transfer overlay area. This field will be modified by the Device Controller to next dTD pointer during endpoint priming or queue advance.
4-0	Reserved. Bit reserved for future use and should be set to zero.

#### 52.4.5.1.4 Set-up Buffer

The set-up buffer is dedicated storage for the 8-byte data that follows a set-up PID.

#### NOTE

Each endpoint has a TX and an RX dQH associated with it, and only the RX queue head is used for receiving setup data packets.

The following table describes the Multiple Mode Control.

**Table 52-59. Multiple Mode Control (HCCPARAMS)**

DWord	Bits	Description
1	31-0	Setup Buffer 0. This buffer contains bytes 3 to 0 of an incoming setup buffer packet and is written by the device controller to be read by software.
2	31-0	Setup Buffer 1. This buffer contains bytes 7 to 4 of an incoming setup buffer packet and is written by the device controller to be read by software.

### 52.4.5.2 Endpoint Transfer Descriptor (dTD)

The dTD describes to the device controller the location and quantity of data to be sent/received for a given transfer.

The DCD should not attempt to modify any field in an active dTD except the Next Like Pointer, which should only be modified as described in section [Managing Transfers with Transfer Descriptors](#).

Table below shows the Endpoint Transfer Descriptor (dTD).

**Table 52-60. Endpoint Transfer Descriptor (dTD)**

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0						
Next Link Pointer																										0		T									
0	Total Bytes															ioc	0		MultO	0		Status															
Buffer Pointer (Page 0)																				Current Offset																	

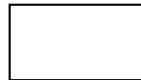
*Table continues on the next page...*

**Table 52-60. Endpoint Transfer Descriptor (dTD) (continued)**

Buffer Pointer (Page 1)	0	Frame Number
Buffer Pointer (Page 2)	Reserved	
Buffer Pointer (Page 3)	Reserved	
Buffer Pointer (Page 4)	Reserved	



Host Controller Read/Write



Host Controller Read Only

The following table describes the dTD Pointer.

**Table 52-61. Next dTD Pointer**

Bit	Description
31-5	Next Transfer Element Pointer. This field contains the physical memory address of the next dTD to be processed. The field corresponds to memory address signals [31:5], respectively.
4-1	Reserved. Bits reserved for future use and should be set to zero.
0	Terminate (T). 1=pointer is invalid. 0=Pointer is valid (points to a valid Transfer Element Descriptor). This bit indicates to the Device Controller that there are no more valid entries in the queue.

The following table describes the dTD Token.

**Table 52-62. dTD Token**

Bit	Description
31	Reserved. Bit reserved for future use and should be set to zero.
30-16	<p>Total Bytes. This field specifies the total number of bytes to be moved with this transfer descriptor. This field is decremented by the number of bytes actually moved during the transaction and only on the successful completion of the transaction.</p> <p>The maximum value software may store in the field is 5*4K(5000H). This is the maximum number of bytes 5 page pointers can access. Although it is possible to create a transfer up to 20K this assumes the 1<sup>st</sup> offset into the first page is 0. When the offset cannot be predetermined, crossing past the 5th page can be guaranteed by limiting the total bytes to 16K**. Therefore, the maximum recommended transfer is 16K(4000H).</p> <p>If the value of the field is zero when the host controller fetches this transfer descriptor (and the active bit is set), the device controller executes a zero-length transaction and retires the transfer descriptor.</p> <p>It is not a requirement for IN transfers that Total Bytes To Transfer be an even multiple of <i>Maximum Packet Length</i>. If software builds such a transfer descriptor for an IN transfer, the last transaction will always be less than <i>Maximum Packet Length</i>.</p>
15	Interrupt On Complete (IOC). This bit is used to indicate if USBINT is to be set in response to device controller being finished with this dTD.
14-12	Reserved. Bits reserved for future use and should be set to zero.

*Table continues on the next page...*

**Table 52-62. dTD Token (continued)**

11-10	<p>Multiplier Override (MultiO). This field can be used for transmit ISO's (ie. ISO-IN) to override the multiplier in the QH. This field must be zero for all packet types that are not transmit-ISO.</p> <p>Example:</p> <p>if QH.multiplier = 3; Maximum packet size = 8; Total Bytes = 15; MultiO = 0 [default]</p> <p>Three packets are sent: {Data2(8); Data1(7); Data0(0)}</p> <p>if QH.multiplier = 3; Maximum packet size = 8; Total Bytes = 15; MultiO = 2</p> <p>Two packets are sent: {Data1(8); Data0(7)}</p> <p>For maximal efficiency, software should compute MultiO = greatest integer of (Total Bytes / Max. Packet Size) except for the case when Total Bytes = 0; then MultiO should be 1.</p> <p>Note: Non-ISO and Non-TX endpoints must set MultiO="00".</p>
9-8	Reserved. Bits reserved for future use and should be set to zero.
7-0	<p>Status. This field is used by the Device Controller to communicate individual command execution states back to the Device Controller software. This field contains the status of the last transaction performed on this qTD. The bit encodings are:</p> <p>Bit Status Field Description 7 Active. 6 Halted. 5 Data Buffer Error. 3 Transaction Error. 4,2,0 Reserved.</p>

The table below describes the dTD Buffer Page Pointer List.

**Table 52-63. dTD Buffer Page Pointer List**

Bit	Description
31-12	Buffer Pointer. Selects the page offset in memory for the packet buffer. Non virtual memory systems will typically set the buffer pointers to a series of incrementing integers.
0,11-0	Current Offset. Offset into the 4kb buffer where the packet is to begin.
1,10-0	Frame Number. Written by the device controller to indicate the frame number in which a packet finishes. This is typically be used to correlate relative completion times of packets on an ISO endpoint.

## 52.4.6 Device Operational Model

The function of the device operation is to transfer a request in the memory image to and from the Universal Serial Bus.

Using a set of linked list transfer descriptors, pointed to by a queue head, the device controller will perform the data transfers. The following sections explain the use of the device controller from the device controller driver (DCD) point-of-view and further describe how specific USB bus events relate to status changes in the device controller programmer's interface.

### 52.4.6.1 Device Controller Initialization

After hardware reset, the device is disabled until the Run/Stop bit is set to a '1'. In the disabled state, the pull-up on the USB D+ is not active which prevents an attach event from occurring. At a minimum, it is necessary to have the queue heads setup for endpoint zero before the device attach occurs.

Shortly after the device is enabled, a USB reset will occur followed by setup packet arriving at endpoint 0. A Queue head must be prepared so that the device controller can store the incoming setup packet.

In order to initialize a device, the software should perform the following steps:

- Set Controller Mode in the USB.USBMODE register to device mode.

#### NOTE

Transitioning from host mode to device mode requires a device controller reset before modifying USB.USBMODE.

- Allocate and Initialize device queue heads in system memory.
  - Minimum: Initialize device queue heads 0 Tx & 0 Rx.

#### NOTE

All device queue heads for control endpoints must be initialized before the endpoint is enabled. Non-Control device queue heads before the endpoint can be used.

- For information on device queue heads, refer to section [Device Data Structures](#).
- Configure USB.ENDPOINTLISTADDR Pointer.
  - For additional information on USB.ENDPOINTLISTADDR, refer to the register table.
- Enable the microprocessor interrupt associated with the USB core.
  - Recommended: enable all device interrupts including: USBINT, USBERRINT, Port Change Detect, USB Reset Received, DCSuspend.
  - For a list of available interrupts refer to the [Interrupt Enable Register \(USBC\\_n\\_USBINTR\)](#) and the [USB Status Register \(USBC\\_n\\_USBSTS\)](#) register tables.
- Set Run/Stop bit to Run Mode.
  - After the Run bit is set and the device is connected to a host, a Bus Reset will be issued by host downstream port. The DCD must monitor the reset event and adjust the software state as described in the Bus Reset section of the Port State and Control section below.

**NOTE**

Endpoint 0 is designed as a control endpoint only and does not need to be configured using ENDPTCTRL0 register.

It is also not necessary to prime Endpoint 0 initially because the first packet received will always be a setup packet. The contents of the first setup packet will require a response in accordance with USB device framework (Chapter 9) command set.

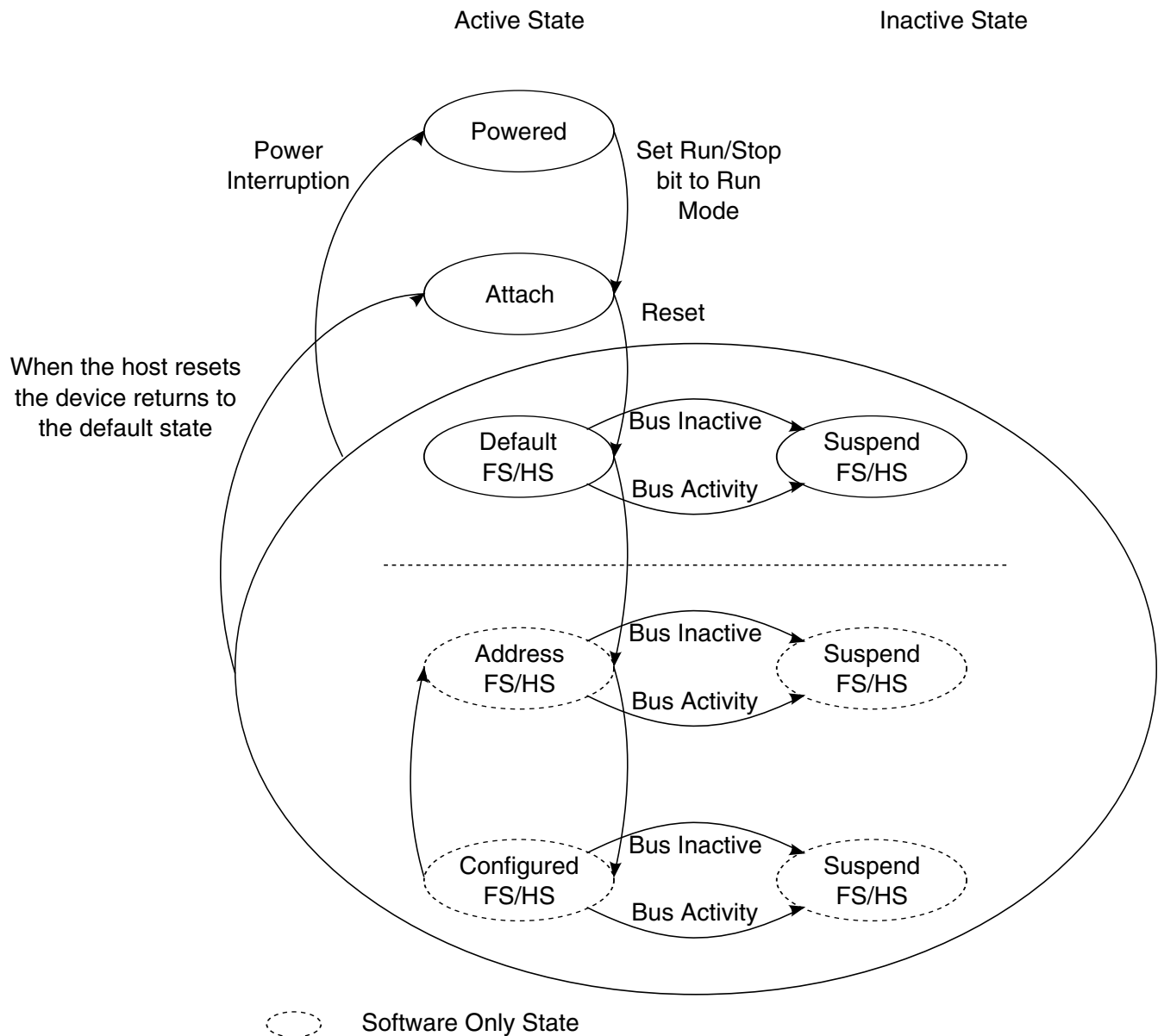
**52.4.6.2 Port State and Control**

From a chip or system reset, the device controller enters the *powered* state. A transition from the *powered* state to the *attach* state occurs when the Run/Stop bit is set to a '1'.

After receiving a reset on the bus, the port will enter the *defaultFS* or *defaultHS* state in accordance with the reset protocol described in Appendix C.2 of the USB Specification Rev. 2.0.

The following state diagram depicts the state of a USB 2.0 device.



**Figure 52-29. Device State Diagram**

States *powered*, *attach*, *defaultFS/HS*, *suspendFS/HS* are implemented in the device controller and are communicated to the DCD using the following status bits:

The following table describes the Device Controller State Information Bits.

**Table 52-64. Device Controller State Information Bits**

Bit	Register
DCSuspend	USB Status Register (USBC_n_USBSTS)
USB Reset Received	USB Status Register (USBC_n_USBSTS)
Port Change Detect	USB Status Register (USBC_n_USBSTS)
High-Speed Port	Port Status & Control (USBC_n_PORTSC1)

It is the responsibility of the DCD to maintain a state variable to differentiate between the *DefaultFS/HS* state and the *Address/Configured* states. Change of state from *Default* to *Address* and the *Configured* states is part of the enumeration process described in the device framework section of the USB 2.0 Specification.

As a result of entering the *Address* state, the device address register (DEVICEADDR) must be programmed by the DCD.

Entry into the *Configured* indicates that all endpoints to be used in the operation of the device have been properly initialized by programming the USB\_UOG\_ENDPTCTRLx registers and initializing the associated queue heads.

#### 52.4.6.2.1 Bus Reset

A bus reset is used by the host to initialize downstream devices.

When a bus reset is detected, the device controller will renegotiate its attachment speed, reset the device address to 0, and notify the DCD by interrupt (assuming the USB Reset Interrupt Enable is set). After a reset is received, all endpoints (except endpoint 0) are disabled and any primed transactions will be cancelled by the device controller. The concept of priming will be clarified below, but the DCD must perform the following tasks when a reset is received:

Clear all setup token semaphores by reading the [Endpoint Status \(USBC\\_n\\_ENDPTSTAT\)](#) register and writing the same value back to the [Endpoint Status \(USBC\\_n\\_ENDPTSTAT\)](#) register.

Clear all the endpoint complete status bits by reading the [Endpoint Complete \(USBC\\_n\\_ENDPTCOMPLETE\)](#) register and writing the same value back to the [Endpoint Complete \(USBC\\_n\\_ENDPTCOMPLETE\)](#) register.

Cancel all primed status by waiting until all bits in the [Endpoint Prime \(USBC\\_n\\_ENDPTPRIME\)](#) are 0 and then writing 0xFFFFFFFF to [Endpoint Flush \(USBC\\_n\\_ENDPTFLUSH\)](#).

Read the reset bit in the [Port Status & Control \(USBC\\_n\\_PORTSC1\)](#) register and make sure that it is still active. A USB reset will occur for a minimum of 3 ms and the DCD must reach this point in the reset cleanup before end of the reset occurs, otherwise a hardware reset of the device controller is recommended (rare.)

- A hardware reset can be performed by writing a one to the device controller reset bit in the USB\_CMD reset. Note: a hardware reset will cause the device to detach from the bus by clearing the Run/Stop bit. Thus, the DCD must completely re-initialize the device controller after a hardware reset.

Free all allocated dTDs because they will no longer be executed by the device controller. If this is the first time the DCD is processing a USB reset event, then it is likely that no dTDs have been allocated.

At this time, the DCD may release control back to the OS because no further changes to the device controller are permitted until a Port Change Detect is indicated.

After a Port Change Detect, the device has reached the default state and the DCD can read the [Port Status & Control \(USBC\\_n\\_PORTSC1\)](#) to determine if the device is operating in FS or HS mode. At this time, the device controller has reached normal operating mode and DCD can begin enumeration according to the USB Chapter 9 - Device Framework.

### NOTE

The device DCD may use the FS/HS mode information to determine the bandwidth mode of the device

In some applications, it may not be possible to enable one or more pipes while in FS mode. *Beyond the data rate issue, there is no difference in DCD operation between FS and HS modes.*

## 52.4.6.2.2 Suspend/Resume

### 52.4.6.2.2.1 Suspend

#### Suspend Description

In order to conserve power, USB devices automatically enter the suspended state when the device has observed no bus traffic for a specified period. When suspended, the USB device maintains any internal status, including its address and configuration. Attached devices must be prepared to suspend at any time they are powered, regardless of if they have been assigned a non-default address, are configured, or neither. Bus activity may cease due to the host entering a suspend mode of its own. In addition, a USB device shall also enter the suspended state when the hub port it is attached to is disabled.

A USB device exits suspend mode when there is bus activity. A USB device may also request the host to exit suspend mode or selective suspend by using electrical signaling to indicate remote wakeup. The ability of a device to signal remote wakeup is optional. If the USB device is capable of remote wakeup signaling, the device must support the ability of the host to enable and disable this capability. When the device is reset, remote wakeup signaling must be disabled.

#### Suspend Operational Model

The device controller moves into the suspend state when suspend signaling is detected or activity is missing on the upstream port for more than a specific period. After the device controller enters the suspend state, the DCD is notified by an interrupt (assuming *DC Suspend Interrupt* is enabled). When the *DCSuspend* bit in the [Port Status & Control \(USBC\\_n\\_PORTSC1\)](#) is set to a '1', the device controller is suspended.

DCD response when the device controller is suspended is application specific and may involve switching to low power operation.

Information on the bus power limits in suspend state can be found in USB 2.0 specification.

#### NOTE

Review system level clocking issues defined in section (Ref: Signals-Clocking) for the clocking requirements of a suspended device controller.

#### 52.4.6.2.2 Resume

If the device controller is suspended, its operation is resumed when any non-idle signaling is received on its upstream facing port. In addition, the device can signal the system to resume operation by forcing resume signaling to the upstream port.

Resume signaling is sent upstream by writing a '1' to the Resume bit in the in the [Port Status & Control \(USBC\\_n\\_PORTSC1\)](#) while the device is in suspend state. Sending resume signal to an upstream port should cause the host to issue resume signaling and bring the suspended bus segment (one more devices) back to the active condition.

#### NOTE

Before resume signaling can be used, the host must enable it by using the Set Feature command defined in device framework (chapter 9) of the USB 2.0 Specification.

#### 52.4.6.2.3 Managing Endpoints

The USB 2.0 specification defines an endpoint, also called a device endpoint or an address endpoint as a uniquely addressable portion of a USB device that can source or sink data in a communications channel between the host and the device.

The endpoint address is specified by the combination of the endpoint number and the endpoint direction.

The channel between the host and an endpoint at a specific device represents a data pipe. Endpoint 0 for a device is always a *control* type data channel used for device discovery and enumeration. Other types of endpoints support by USB include *bulk*, *interrupt*, and

*isochronous*. Each endpoint type has specific behavior related to packet response and error handling. More detail on endpoint operation can be found in the USB 2.0 specification.

The USB OTG device controller hardware supports up to 8 endpoint numbers.

Each endpoint direction is essentially independent and can be configured with differing behavior in each direction. For example, the DCD can configure endpoint 1-IN to be a bulk endpoint and endpoint 1-OUT to be an isochronous endpoint. This helps to conserve the total number of endpoints required for device operation. The only exception is that control endpoints must use both directions on a single endpoint number to function as a control endpoint. Endpoint 0 is, for example, is always a control endpoint and uses the pair of directions.

Each endpoint direction requires a *queue head* allocated in memory. To support the 8 endpoint numbers, 16 *queue heads* are required. The operation of an endpoint and use of *queue heads* are described later in this document.

#### 52.4.6.2.4 Endpoint Initialization

After hardware reset, all endpoints except endpoint zero are uninitialized and disabled. The DCD must configure and enable each endpoint by writing to configuration bit in the USB\_UOG\_ENDPTCTRLx register.

Each 32-bit USB\_UOG\_ENDPTCTRLx is split into an upper and lower half. The lower half of USB\_UOG\_ENDPTCTRLx is used to configure the receive or OUT endpoint and the upper half is likewise used to configure the corresponding transmit or IN endpoint. Control endpoints must be configured the same in both the upper and lower half of the USB\_UOG\_ENDPTCTRLx register otherwise the behavior is undefined. The following table shows how to construct a configuration word for endpoint initialization. The following table shows the fields and values for the Device Controller Endpoint initialization.

**Table 52-65. Device Controller Endpoint Initialization**

Field	Value
Data Toggle Reset	1
Data Toggle Inhibit	0
Endpoint Type	00 Control 01 Isochronous 10 Bulk 11 Interrupt
Endpoint Stall	0

### 52.4.6.2.5 Stalling

There are two occasions where the device controller may need to return to the host a STALL.

The first occasion is the functional stall, which is a condition set by the DCD as described in the USB 2.0 device framework. A functional stall is only used on non-control endpoints and can be enabled in the device controller by setting the endpoint stall bit in the USB\_UOG\_ENDPTCTRLx register associated with the given endpoint and the given direction. In a functional stall condition, the device controller will continue to return STALL responses to all transactions occurring on the respective endpoint and direction until the endpoint stall bit is cleared by the DCD.

A protocol stall, unlike a function stall, is used on control endpoints is automatically cleared by the device controller at the start of a new control transaction (setup phase). When enabling a protocol stall, the DCD should enable the stall bits (both directions) as a pair. A single write to the USB\_UOG\_ENDPTCTRLx register can ensure that both stall bits are set at the same instant.

#### NOTE

Any write to the USB\_UOG\_ENDPTCTRLx register during operational mode must preserve the endpoint type field (that is, perform a read-modify-write).

The following table shows the response matrix for the Device Controller Stall.

**Table 52-66. Device Controller Stall Response Matrix**

USB Packet	Endpoint Stall Bit.	Effect on STALL bit.	USB Response
SETUP packet received by a non-control endpoint.	N/A	None.	STALL
IN/OUT/PING packet received by a non-control endpoint.	'1'	None.	STALL
IN/OUT/PING packet received by a non-control endpoint.	'0'	None.	ACK/ NAK/ NYET
SETUP packet received by a control endpoint.	N/A	Cleared	ACK
IN/OUT/PING packet received by a control endpoint	'1'	None	STALL
IN/OUT/PING packet received by a control endpoint.	'0'	None.	ACK/ NAK/ NYET

### 52.4.6.2.6 Data Toggle

Data toggle is a mechanism to maintain data coherency between host and device for any given data pipe.

For more information on data toggle, refer to the USB 2.0 specification.

#### 52.4.6.2.6.1 Data Toggle Reset

The DCD may reset the data toggle state bit and cause the data toggle sequence to reset in the device controller by writing a '1' to the data toggle reset bit in the USB\_UOG\_ENDPTCTRLx register.

This should only be necessary when configuring/initializing an endpoint or returning from a STALL condition.

#### 52.4.6.2.6.2 Data Toggle Inhibit

##### NOTE

This feature is for test purposes only and should never be used during normal device controller operation.

Setting the *data toggle Inhibit bit* active ('1') causes the device controller to ignore the data toggle pattern that is normally sent and accept all incoming data packets regardless of the data toggle state.

In normal operation, the device controller checks the DATA0/DATA1 bit against the data toggle to determine if the packet is valid. If Data PID does not match the data toggle state bit maintained by the device controller for that endpoint, the Data toggle is considered not valid. If the data toggle is not valid, the device controller assumes the packet was already received and discards the packet (not reporting it to the DCD). To prevent the host controller from re-sending the same packet, the device controller will respond to the error packet by acknowledging it with either an ACK or NYET response.

#### 52.4.6.2.6.3 Priming Transmit Endpoints

Priming a transmit endpoint will cause the device controller to fetch the device transfer descriptor (dTD) for the transaction pointed to by the device queue head (dQH).

After the dTD is fetched, it will be stored in the dQH until the device controller completes the transfer described by the dTD. Storing the dTD in the dQH allows the device controller to fetch the operating context needed to handle a request from the host without the need to follow the linked list, starting at the dQH when the host request is received.

After the device has loaded the dTD, the leading data in the packet is stored in a FIFO in the device controller. This FIFO is split into virtual channels so that the leading data can be stored for any endpoint up to the maximum number of endpoints configured at device synthesis time.

After a priming request is complete, an endpoint state of primed is indicated in the USB\_UOG\_ENDPTSTATUS register. For a primed transmit endpoint, the device controller can respond to an IN request from the host and meet the stringent bus turnaround time of High Speed USB.

Because only the leading data is stored in the device controller FIFO, it is necessary for the device controller to begin filling in behind leading data after the transaction starts. The FIFO must be sized to account for the maximum latency that can be incurred by the system memory bus. More information about FIFO sizing is presented in section .

#### **52.4.6.2.6.4 Priming Receive Endpoints**

Priming receive endpoints is identical to priming of transmit endpoints from the point of view of the DCD. At the device controller the major difference in the operational model is that there is no data movement of the leading packet data simply because the data is to be received from the host.

Note as part of the architecture, the FIFO for the receive endpoints is not partitioned into multiple channels like the transmit FIFO. Thus, the size of the RX FIFO does not scale with the number of endpoints.

### **52.4.6.3 Operational Model For Packet Transfers**

All transactions on the USB bus are initiated by the host and in turn, the device must respond to any request from the host within the turnaround time stated in the USB 2.0 Specification.

At USB 1.1 Full or Low Speed rates, this turnaround time was significant and the USB 1.1 device controllers were architected so that the device controller could access main memory or interrupt a host protocol processor in order to respond to the USB 1.1 transaction. The architecture of the USB 2.0 device controller must be different because same methods will not meet USB 2.0 High-speed turnaround time requirements by simply increasing clock rate.

A USB host will send requests to the device controller in an order that can not be precisely predicted as a single pipeline, so it is not possible to prepare a single packet for the device controller to execute. However, the order of packet requests is predictable when the endpoint number and direction is considered. For example, if endpoint 3



(transmit direction) is configured as a bulk pipe, then we can expect the host will send IN requests to that endpoint. This device controller is architected in such a way that it can prepare packets for each endpoint/direction in anticipation of the host request. The process of preparing the device controller to send or receive data in response to host initiated transaction on the bus is referred to as "priming" the endpoint. This term will be used throughout the following documentation to describe the device controller operation so the DCD can be architected properly use priming. Further, note that the term "flushing" is used to describe the action of clearing a packet that was queued for execution.

#### 52.4.6.3.1 Interrupt/Bulk Endpoint Operational Model

The behaviors of the device controller for interrupt and bulk endpoints are identical.

All valid IN and OUT transactions to bulk pipes will handshake with a NAK unless the endpoint had been primed. Once the endpoint has been primed, data delivery will commence.

A dTD will be retired by the device controller when the packets described in the transfer descriptor have been completed. Each dTD describes N packets to be transferred according to the USB Variable Length transfer protocol. The formula and table on the following page describe how the device controller computes the number and length of the packets to be sent/received by the USB vary according to the total number of bytes and maximum packet length.

With Zero Length Termination (ZLT) = 0

$$N = \text{INT}(\text{Number Of Bytes}/\text{Max. Packet Length}) + 1$$

With Zero Length Termination (ZLT) = 1

$$N = \text{MAXINT}(\text{Number Of Bytes}/\text{Max. Packet Length})$$

**Table 52-67. Variable Length Transfer Protocol Example (ZLT = 0)**

Bytes (dTD)	Max. Packet Length (dQH)	N	P1	P2	P3
511	256	2	256	255	
512	256	3	256	256	0
512	512	2	512	0	

**Table 52-68. Variable Length Transfer Protocol Example (ZLT = 1)**

Bytes (dTD)	Max. Packet Length (dQH)	N	P1	P2	P3
511	256	2	256	255	
512	256	2	256	256	
512	512	1	512		

**NOTE**

The MULT field in the dQH must be set to "00" for bulk, interrupt, and control endpoints.

TX-dTD is complete when:

- All packets described dTD were successfully transmitted. \*\*\* Total bytes in dTD will equal zero when this occurs.

RX-dTD is complete when:

- All packets described in dTD were successfully received. \*\*\* Total bytes in dTD will equal zero when this occurs.
- A short packet (number of bytes < maximum packet length) was received. \*\*\* This is a successful transfer completion; DCD must check Total Bytes in dTD to determine the number of bytes that are remaining. From the total bytes remaining in the dTD, the DCD can compute the actual bytes received.
- A long packet was received (number of bytes > maximum packet size) OR (total bytes received > total bytes specified). \*\*\* This is an error condition. The device controller will discard the remaining packet, and set the Buffer Error bit in the dTD. In addition, the endpoint will be flushed and the USBERR interrupt will become active.

On the successful completion of the packet(s) described by the dTD, the active bit in the dTD will be cleared and the next pointer will be followed when the Terminate bit is clear. When the Terminate bit is set, the device controller will flush the endpoint/direction and cease operations for that endpoint/direction.

On the unsuccessful completion of a packet (see long packet above), the dQH will be left pointing to the dTD that was in error. In order to recover from this error condition, the DCD must properly reinitialize the dQH by clearing the active bit and update the nextTD pointer before attempting to re-prime the endpoint.

**NOTE**

All packet level errors such as a missing handshake or CRC error will be retried automatically by the device controller.

There is no required interaction with the DCD for handling such errors.

**52.4.6.3.1.1 Interrupt/Bulk Endpoint Bus Response Matrix**

The table below shows the response matrix for Interrupt/Bulk Endpoint Bus.

**Table 52-69. Interrupt/Bulk Endpoint Bus Response Matrix**

	Stall	Not Primed	Primed	Underflow	Overflow
Setup	Ignore	Ignore	Ignore	N/A	N/A
In	STALL	NAK	Transmit	BS Error	N/A
Out	STALL	NAK	Receive + NYET/ACK	N/A	NAK
Ping	STALL	NAK	ACK	N/A	N/A
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore

**NOTE**

BS Error = Force Bit Stuff Error

NYET/ACK - NYET unless the Transfer Descriptor has packets remaining according to the USB variable length protocol then ACK.

SYSERR - System error should never occur when the latency FIFOs are correctly sized and the DCD is responsive.

**52.4.6.3.2 Control Endpoint Operation Model****52.4.6.3.2.1 Setup Phase**

All requests to a control endpoint begin with a setup phase followed by an optional data phase and a required status phase. The device controller will always accept the setup phase unless the setup lockout is engaged.

The setup lockout will engage so that future setup packets are ignored. Lockout of setup packets ensures that while software is reading the setup packet stored in the queue head, that data is not written as it is being read potentially causing an invalid setup packet.

The setup lockout mechanism can be disabled and a new tripwire type semaphore will ensure that the setup packet payload is extracted from the queue head without being corrupted by an incoming setup packet. This is the preferred behavior because ignoring repeated setup packets due to long software interrupt latency would be a compliance issue.

- Disable Setup Lockout by writing 1 to Setup Lockout Mode (SLOM) in [USB Device Mode \(USBC\\_n\\_USBMODE\)](#). (once at initialization). Setup lockout is not necessary when using the tripwire as described below.

**NOTE**

Leaving the Setup Lockout Mode As 0 will result in pre-2.3 hardware behavior.

- After receiving an interrupt and inspecting [Endpoint Setup Status \(USBC\\_n\\_ENDPTSETUPSTAT\)](#) to determine that a setup packet was received on a particular pipe:
  - a. Write 1 to clear corresponding bit [Endpoint Setup Status \(USBC\\_n\\_ENDPTSETUPSTAT\)](#).
  - b. Write 1 to Setup Tripwire (SUTW) in [USB Command Register \(USBC\\_n\\_USBCMD\)](#) register.
  - c. Duplicate contents of dQH.SetupBuffer into local software byte array.
  - d. Read Setup TripWire (SUTW) in [USB Command Register \(USBC\\_n\\_USBCMD\)](#) register. (if set - continue; if cleared - goto 2)
  - e. Write 0 to clear Setup Tripwire (SUTW) in [USB Command Register \(USBC\\_n\\_USBCMD\)](#) register.
  - f. Process setup packet using local software byte array copy and execute status/handshake phases.

### NOTE

After receiving a new setup packet the status and/or handshake phases may still be pending from a previous control sequence. These should be flushed & deallocated before linking a new status and/or handshake dTD for the most recent setup packet.

#### 52.4.6.3.2.2 Data Phase

Following the setup phase, the DCD must create a device transfer descriptor for the data phase and prime the transfer.

After priming the packet, the DCD must verify a new setup packet has not been received by reading the USB.ENDPTSETUPSTAT register immediately verifying that the prime had completed. A prime will complete when the associated bit in the [Endpoint Prime \(USBC\\_n\\_ENDPTPRIME\)](#) register is zero and the associated bit in the [Endpoint Status \(USBC\\_n\\_ENDPTSTAT\)](#) register is a one. If a prime fails, ie. The [Endpoint Prime \(USBC\\_n\\_ENDPTPRIME\)](#) bit goes to zero and the [Endpoint Status \(USBC\\_n\\_ENDPTSTAT\)](#) bit is not set, then the prime has failed. This can only be due to improper setup of the dQH, dTD or a setup arriving during the prime operation. If a new setup packet is indicated after the ENDPTPRIME bit is cleared, then the transfer descriptor can be freed and the DCD must reinterpret the setup packet.

Should a setup arrive after the data stage is primed, the device controller will automatically clear the prime status ([Endpoint Status \(USBC\\_n\\_ENDPTSTAT\)](#)) to enforce data coherency with the setup packet.

**NOTE**

- The MULT field in the dQH must be set to "00" for bulk, interrupt, and control endpoints.
- Error handling of data phase packets is the same as bulk packets described previously.

**52.4.6.3.2.3 Status Phase**

Similar to the data phase, the DCD must create a transfer descriptor (with byte length equal zero) and prime the endpoint for the status phase.

The DCD must also perform the same checks of the USB.ENDPTSETUPSTAT as described above in the data phase.

**NOTE**

- The MULT field in the dQH must be set to 00 for bulk, interrupt, and control endpoints.
- Error handling of data phase packets is the same as bulk packets described previously.

**52.4.6.3.2.4 Control Endpoint Bus Response Matrix**

Shown in the following table is the device controller response to packets on a control endpoint according to the device controller state.

The table below shows the response matrix for the Control Endpoint Bus.

**Table 52-70. Control Endpoint Bus Response Matrix**

Token Type	Endpoint State					Setup Lockout
	Stall	Not Primed	Primed	Underflow	Overflow	
Setup	ACK	ACK	ACK	N/A	SYSERR	
In	STALL	NAK	Transmit	BS Error	N/A	N/A
Out	STALL	NAK	Receive + NYET/ ACK	N/A	NAK	N/A
Ping	STALL	NAK	ACK	N/A	N/A	N/A
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore	Ignore

BS Error = Force Bit Stuff Error

NYET/ACK - NYET unless the Transfer Descriptor has packets remaining according to the USB variable length protocol then ACK.

SYSERR - System error should never occur when the latency FIFOs are correctly sized and the DCD is responsive.

### 52.4.6.3.3 Isochronous Endpoint Operational Model

Isochronous endpoints are used for real-time scheduled delivery of data and their operational model is significantly different than the host throttled Bulk, Interrupt, and Control data pipes.

Real time delivery by the device controller will be accomplished by the following:

- Exactly MULT Packets per (micro)Frame are transmitted/received. Note: MULT is a two-bit field in the device Queue Head. The variable length packet protocol is not used on isochronous endpoints.
- NAK responses are not used. Instead, zero length packets are sent in response to an IN request to an unprimed endpoints. For unprimed RX endpoints, the response to an OUT transaction is to ignore the packet within the device controller.
- Prime requests always schedule the transfer described in the dTD for the next (micro)frame. If the ISO-dTD is still active after that frame, then the ISO-dTD will be held ready until executed or canceled by the DCD.

An EHCI compatible host controller uses the periodic frame list to schedule data exchanges to Isochronous endpoints. The operational model for device mode does not use such a data structure. Instead, the same dTD used for Control/Bulk/Interrupt endpoints is also used for isochronous endpoints. The difference is in the handling of the dTD.

The first difference between bulk and ISO-endpoints is that priming an ISO-endpoint is a delayed operation such that an endpoint will become primed only after a SOF is received. After the DCD writes the prime bit, the prime bit will be cleared as usual to indicate to software that the device controller completed a priming the dTD for transfer. Internal to the design, the device controller hardware masks that prime start until the next frame boundary. This behavior is hidden from the DCD but occurs so that the device controller can match the dTD to a specific (micro)frame.

Another difference with isochronous endpoints is that the transaction must wholly complete in a (micro)frame. Once an ISO transaction is started in a (micro)frame it will retire the corresponding dTD when MULT transactions occur or the device controller finds a fulfillment condition.

The transaction error bit set in the status field indicates a fulfillment error condition. When a fulfillment error occurs, the frame after the transfer failed to complete wholly, the device controller will force retire the ISO-dTD and move to the next ISO-dTD.

It is important to note that fulfillment errors are only caused due to partially completed packets. If no activity occurs to a primed ISO-dTD, the transaction will stay primed indefinitely. This means it is up to software discard transmit ISO-dTDs that pile up from a failure of the host to move the data.

Finally, the last difference with ISO packets is in the data level error handling. When a CRC error occurs on a received packet, the packet is not retried similar to bulk and control endpoints. Instead, the CRC is noted by setting the *Transaction Error* bit and the data is stored as usual for the application software to sort out.

- TX Packet Retired
  - MULT counter reaches zero.
  - Fulfillment Error [*Transaction Error* bit is set]
    - # Packets Occurred > 0 AND # Packets Occurred < MULT

#### NOTE

For TX-ISO, MULT Counter can be loaded with a lesser value in the dTD Multiplier Override field in hardware versions 2.3 and later. If the Multiplier Override is zero, the MULT Counter is initialized to the Multiplier in the QH.

- RX Packet Retired:
  - MULT counter reaches zero.
  - Non-MDATA Data PID is received\*\*
    - \*\* Exit criteria only valid in hardware version 2.3 or later. Previous to hardware version 2.3, any PID sequence that did not match the MULT field exactly would be flagged as a transaction error due to PID mismatch or fulfillment error.
  - Overflow Error:
    - Packet received is > maximum packet length. [*Buffer Error* bit is set]
    - Packet received exceeds total bytes allocated in dTD. [*Buffer Error* bit is set]
  - Fulfillment Error [*Transaction Error* bit is set]
    - # Packets Occurred > 0 AND # Packets Occurred < MULT
  - CRC Error [*Transaction Error* bit is set]

#### NOTE

For ISO, when a dTD is retired, the next dTD is primed for the next frame. For continuous (micro)frame to (micro)frame operation the DCD should ensure that the dTD linked-list is out ahead of the device controller by at least two (micro)frames.

#### 52.4.6.3.3.1 Isochronous Pipe Synchronization

When it is necessary to synchronize an isochroous data pipe to the host, the (micro)frame number (USB\_UOG\_FRINDEX register) can be used as a marker.

To cause a packet transfer to occur at a specific (micro)frame number [N], the DCD should interrupt on SOF during frame N-1. When the USB\_UOG\_FRINDEX=N-1, the DCD must write the prime bit. The device controller will prime the isochronous endpoint in (micro)frame N-1 so that the device controller will execute delivery during (micro)frame N.

### NOTE

Priming an endpoint towards the end of (micro)frame N-1 will not guarantee delivery in (micro)frame N. The delivery may actually occur in (micro)frame N+1 if device controller does not have enough time to complete the prime before the SOF for packet N is received.

#### 52.4.6.3.3.2 Isochronous Endpoint Bus Response Matrix

The following table shows the response matrix for the Isochronous Endpoint Bus.

**Table 52-71. Isochronous Endpoint Bus Response Matrix**

	Stall	Not Primed	Primed	Underflow	Overflow
Setup	STALL	STALL	STALL	N/A	N/A
In	NULL Packet	NULL Packet	Transmit	BS Error	N/A
Out	Ignore	Ignore	Receive	N/A	Drop Packet
Ping	Ignore	Ignore	Ignore	Ignore	Ignore
Invalid	Ignore	Ignore	Ignore	Ignore	Ignore

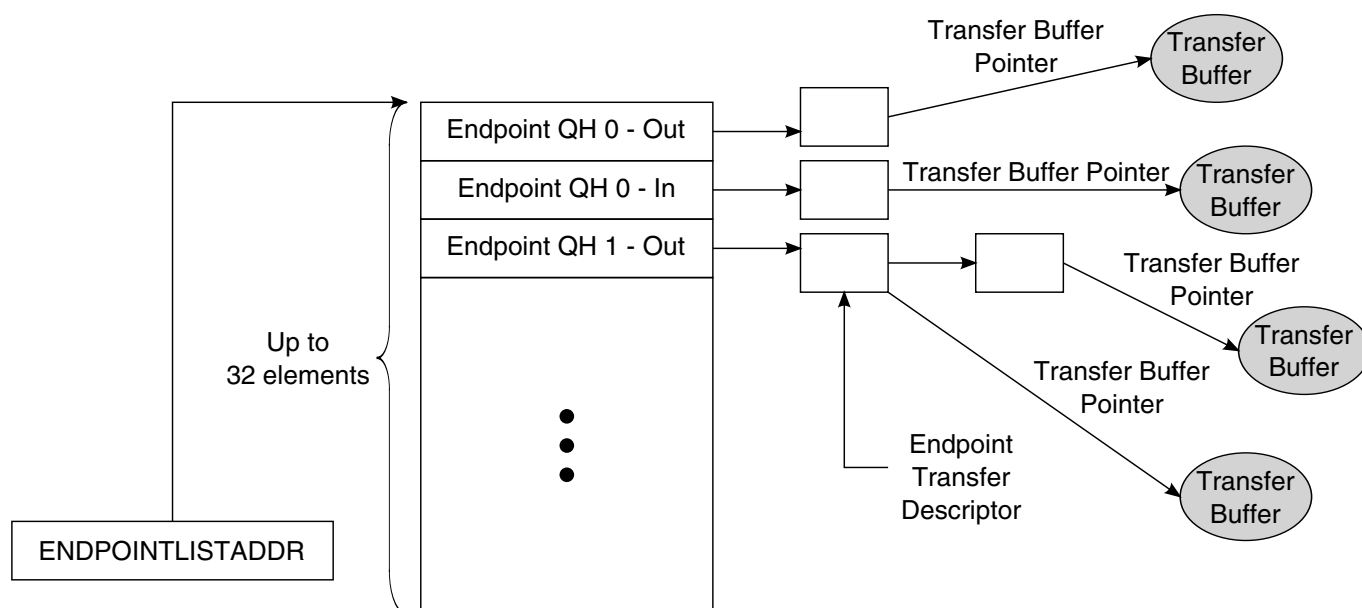
1. BS Error = Force Bit Stuff Error

NULL Packet = Zero Length Packet

#### 52.4.6.4 Managing Queue Heads

The following figure shows the End Point Queue Head.





**Figure 52-30. End Point Queue Head Diagram**

The device queue head (dQH) points to the linked list of transfer tasks, each depicted by the device Transfer Descriptor (dTD). An area of memory pointed to by USB.ENDPOINTLISTADDR contains a group of all dQH's in a sequential list as shown in [Figure 52-30](#). The even elements in the list of dQH's are used for receive endpoints (OUT/SETUP) and the odd elements are used for transmit endpoints (IN/INTERRUPT). Device transfer descriptors are linked head to tail starting at the queue head and ending at a terminate bit. Once the dTD has been retired, it will no longer be part of the linked list from the queue head. Therefore, software is required to track all transfer descriptors because pointers will no longer exist within the queue head once the dTD is retired (see section [Software Link Pointers](#)).

In addition to the current and next pointers and the dTD overlay examined in section [Operational Model For Packet Transfers](#), the dQH also contains the following parameters for the associated endpoint: Multiplier, Maximum Packet Length, Interrupt On Setup. The complete initialization of the dQH including these fields is demonstrated in the next section.

#### 52.4.6.4.1 Queue Head Initialization

One device queue head must be initialized for each active endpoint.

To initialize a device queue head:

- Write the wMaxPacketSize field as required by the USB Chapter 9 or application specific protocol.
- Write the multiplier field to 0 for control, bulk, and interrupt endpoints. For ISO endpoints, set the multiplier to 1,2, or 3 as required bandwidth an in conjunction with the USB Chapter 9 protocol.

**NOTE**

In FS mode, the multiplier field can only be 1 for ISO endpoints.

- Write the next dTD Terminate bit field to 1.
- Write the Active bit in the status field to 0.
- Write the Halt bit in the status field to 0.

**NOTE**

The DCD must only modify dQH if the associated endpoint is not primed and there are no outstanding dTD's.

**52.4.6.4.2 Operational Model For Setup Transfers**

As discussed in section [Control Endpoint Operation Model](#), setup transfer requires special treatment by the DCD. A setup transfer does not use a dTD but instead stores the incoming data from a setup packet in an 8-byte buffer within the dQH.

Upon receiving notification of the setup packet, the DCD should handle the setup transfer as demonstrated here:

1. Copy setup buffer contents from dQH - RX to software buffer.
2. Acknowledge setup backup by writing a "1" to the corresponding bit in ENDPTSETUPSTAT.

**NOTE**

- The acknowledge must occur before continuing to process the setup packet.
  - After the acknowledge has occurred, the DCD must not attempt to access the setup buffer in the dQH - RX. Only the local software copy should be examined.
3. Check for pending data or status dTD's from previous control transfers and flush if any exist as discussed in section [Flushing/De-priming an Endpoint](#).
  4. Decode setup packet and prepare data phase [optional] and status phase transfer as required by the USB Chapter 9 or application specific protocol.

**NOTE**

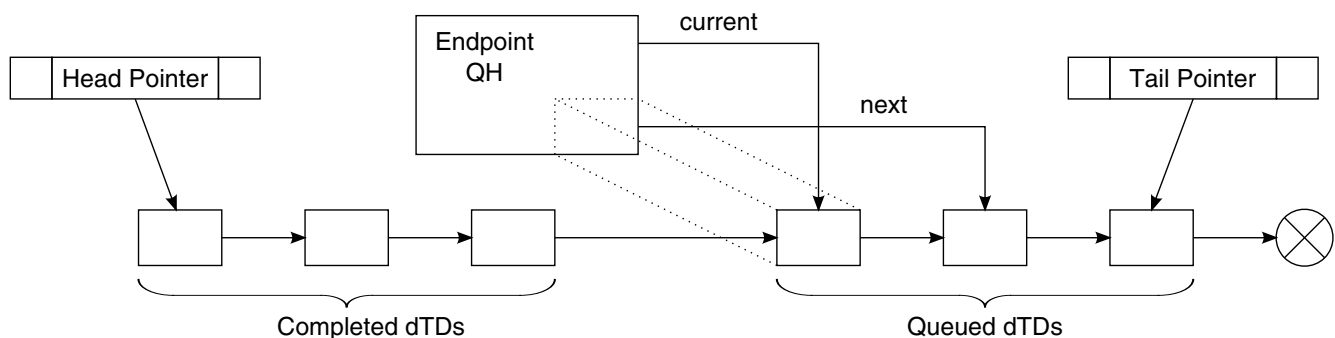
It is possible for the device controller to receive setup packets before previous control transfers complete. Existing control packets in progress must be flushed and the new control packet completed.

## 52.4.6.5 Managing Transfers with Transfer Descriptors

### 52.4.6.5.1 Software Link Pointers

It is necessary for the DCD software to maintain head and tail pointers to the for the linked list of dTDs for each respective queue head.

This is necessary because the dQH only maintains pointers to the current working dTD and the next dTD to be executed. The operations described in next section for managing dTD will assume the DCD can use reference the head and tail of the dTD linked list. The following figure shows the Software Link Pointers.



**Figure 52-31. Software Link Pointers**

**NOTE**

To conserve memory, the reserved fields at the end of the dQH can be used to store the Head & Tail pointers, but it still remains the responsibility of the DCD to maintain the pointers.

### 52.4.6.5.2 Building a Transfer Descriptor

Before a transfer can be executed from the linked list, a dTD must be built to describe the transfer.

Use the following procedure for building dTDs.

Allocate 8-DWord dTD block of memory aligned to 8-DWord boundaries. Example: bit address 4:0 would be equal to "00000"

Write the following fields:

1. Initialize first 7 DWords to 0.
2. Set the terminate bit to 1.
3. Fill in total bytes with transfer size.
4. Set the interrupt on complete if desired.
5. Initialize the status field with the active bit set to 1 and all remaining status bits set to 0.
6. Fill in buffer pointer page 0 and the current offset to point to the start of the data buffer.
7. Initialize buffer pointer page 1 through page 4 to be one greater than each of the previous buffer pointer.

### 52.4.6.5.3 Executing A Transfer Descriptor

To safely add a dTD, the DCD must follow this procedure which will handle the event where the device controller reaches the end of the dTD list at the same time a new dTD is being added to the end of the list.

Determine whether the link list is empty: Check DCD driver to see if pipe is empty (internal representation of linked-list should indicate if any packets are outstanding).

- Case 1: Link list is empty
  - a. Write dQH next pointer AND dQH terminate bit to 0 as a single DWord operation.
  - b. Clear active & halt bit in dQH (in case set from a previous error).
  - c. Prime endpoint by writing 1 to correct bit position in [Endpoint Prime \(USBC\\_n\\_ENDPTPRIME\)](#).
- Case 2: Link list is not empty
  - a. Add dTD to end of linked list.
  - b. Read correct prime bit in [Endpoint Prime \(USBC\\_n\\_ENDPTPRIME\)](#)- if 1 DONE.
  - c. Set ATDTW bit in USBCMD register to 1.
  - d. Read correct status bit in [Endpoint Status \(USBC\\_n\\_ENDPTSTAT\)](#). (store in tmp. variable for later)
  - e. Read ATDTW bit in USBCMD register.
    - If 0 goto 3.
    - If 1 continue to 6.
  - f. Write ATDTW bit in USBCMD register to 0.

- g. If status bit read in (3) is 1 DONE.
- h. If status bit read in (3) is 0 then Goto Case 1: Step 1.

#### 52.4.6.5.4 Transfer Completion

After a dTD has been initialized and the associated endpoint primed the device controller will execute the transfer upon the host-initiated request. The DCD will be notified with a USB interrupt if the Interrupt On Complete bit was set or alternately, the DCD can poll the endpoint complete register to find when the dTD had been executed. After a dTD has been executed, DCD can check the status bits to determine success or failure.

#### NOTE

Multiple dTD can be completed in a single endpoint complete notification. After clearing the notification, DCD must search the dTD linked list and retire all dTDs that have finished (Active bit cleared).

By reading the status fields of the completed dTDs, the DCD can determine if the transfers completed successfully. Success is determined with the following combination of status bits:

- Active = 0
- Halted = 0
- Transaction Error = 0
- Data Buffer Error = 0

Should any combination other than the one shown above exist, the DCD must take proper action. Transfer failure mechanisms are indicated in the [Device Error Matrix](#).

In addition to checking the status bit the DCD must read the Transfer Bytes field to determine the actual bytes transferred. When a transfer is complete, the Total Bytes transferred is by decremented by the actual bytes transferred. For Transmit packets, a packet is only complete after the actual bytes reaches zero, but for receive packets, the host may send fewer bytes in the transfer according the USB variable length packet protocol.

#### 52.4.6.5.5 Flushing/De-priming an Endpoint

It is necessary for the DCD to flush to de-prime one more endpoints on a USB device reset or during a broken control transfer.

There may also be application specific requirements to stop transfers in progress. The following procedure can be used by the DCD to stop a transfer in progress:

1. Write a '1' to the corresponding bit(s) in [Endpoint Flush \(USBC\\_n\\_ENDPTFLUSH\)](#).

2. Wait until all bits in [Endpoint Flush \(USBC\\_n\\_ENDPTFLUSH\)](#) are '0'.
  - Software note: this operation may take a large amount of time depending on the USB bus activity. It is not desirable to have this wait loop within an interrupt service routine.
3. Read [Endpoint Status \(USBC\\_n\\_ENDPTSTAT\)](#) to ensure that for all endpoints commanded to be flushed, that the corresponding bits are now '0'. If the corresponding bits are '1' after step #2 has finished, then the flush failed as described in the following:
  - Explanation: In very rare cases, a packet is in progress to the particular endpoint when commanded flush using [Endpoint Flush \(USBC\\_n\\_ENDPTFLUSH\)](#). A safeguard is in place to refuse the flush to ensure that the packet in progress completes successfully. The DCD may need to repeatedly flush any endpoints that fail to flush by repeating steps 1-3 until each endpoint is successfully flushed.

#### 52.4.6.5.6 Device Error Matrix

The following table summarizes packet errors that are not automatically handled by the Device Controller.

**Table 52-72. Device Error Matrix**

Error	Direction	Packet Type	Data Buffer Error Bit	Transaction Error Bit
Overflow **	RX	Any	1	0
ISO Packet Error	RX	ISO	0	1
ISO Fulfillment Error	Both	ISO	0	1

Notice that the device controller handles all errors on Bulk/Control/Interrupt Endpoints except for a data buffer overflow. However, for ISO endpoints, errors packets are not retried and errors are tagged as indicated. The table below describes the errors.

**Table 52-73. Error Descriptions**

Error	Description
Overflow	Number of bytes received exceeded max. packet size or total buffer length. ** This error will also set the Halt bit in the dQH and if there are dTDs remaining in the linked list for the endpoint, then those will not be executed.
ISO Packet Error	CRC Error on received ISO packet. Contents not guaranteed to be correct.
ISO Fulfillment Error	Host failed to complete the number of packets defined in the dQH mult field within the given (micro)frame. For scheduled data delivery the DCD may need to readjust the data queue because a fulfillment error will cause Device Controller to cease data transfers on the pipe for one (micro)frame. During the "dead" (micro)frame, the Device Controller reports error on the pipe and primes for the following frame.

## 52.4.6.6 Servicing Interrupts

The interrupt service routine must consider that there are high-frequency, low-frequency operations, and error operations and order accordingly.

### 52.4.6.6.1 High-Frequency Interrupts

High frequency interrupts in particular should be handled in the order below. The most important of these is listed first because the DCD must acknowledge a setup buffer in the timeliest manner possible.

The table below describes the High frequency interrupt events.

**Table 52-74. High Frequency Interrupt Events**

Execution Order	Interrupt	Action
1a	USB Interrupt - USB.ENDPTSETUPSTATUS	Copy contents of setup buffer and acknowledge setup packet (as indicated in <a href="#">Figure 52-30</a> shows the End Point Queue Head). Process setup packet according to USB 2.0 Chapter 9 or application specific protocol.
1b	USB Interrupt <sup>1</sup> - USB.ENDPTCOMPLETE	Handle completion of dTD as indicated in <a href="#">Figure 52-30</a> shows the End Point Queue Head.
2	SOF Interrupt	Action as deemed necessary by application. This interrupt may not have a use in all applications.

1. It is likely that multiple interrupts to stack up on any call to the Interrupt Service Routine AND during the Interrupt Service Routine.

### 52.4.6.6.2 Low-Frequency Interrupts

The low frequency interrupts can be handled in any order because they do not occur often in comparison to the high-frequency interrupts.

The table below shows the Low frequency interrupt events.

**Table 52-75. Low Frequency Interrupt Events**

Interrupt	Action
Port Change	Change software state information.
Sleep Enable (Suspend)	Change software state information. Low power handling as necessary.
Reset Received	Change software state information. Abort pending transfers.

### 52.4.6.6.3 Error Interrupts

Error interrupts will be least frequent and should be placed last in the interrupt service routine.

The following table shows the error interrupt events.

**Table 52-76. Error Interrupt Events**

Interrupt	Action
USB Error Interrupt	This error is redundant because it combines USB Interrupt and an error status in the dTD. The DCD will more aptly handle packet-level errors by checking dTD status field upon receipt of USB Interrupt (w/ USB.ENDPTCOMPLETE).
System Error	Unrecoverable error. Immediate Reset of core; free transfers buffers in progress and restart the DCD.

## 52.5 USB Non-Core Memory Map/Register Definition

There are two kinds of registers in the USB module: USB core registers and USB non-core registers.

USB core registers are used to control USB core functions, and more independent of USB features. Each USB controller core has its own core registers.

USB non-core registers are additional to USB core registers, and more dependent on USB features. i.MX series products vary in non-core registers.

This section describes only the USB non-core registers. For detailed descriptions of USB core registers, please refer to [Register Interface](#).

### NOTE

For reserved bits, please preserve the value when writing (read its reset value, then write this value back).

"USB\_UOG1\_", "USB\_UOG2\_", "USB\_UH\_" prefix in register name indicates it is a core register for OTG1/OTG2/Host controller core respectively.

"USB\_USB\_" prefix in register name indicates it is a USB non-core register.



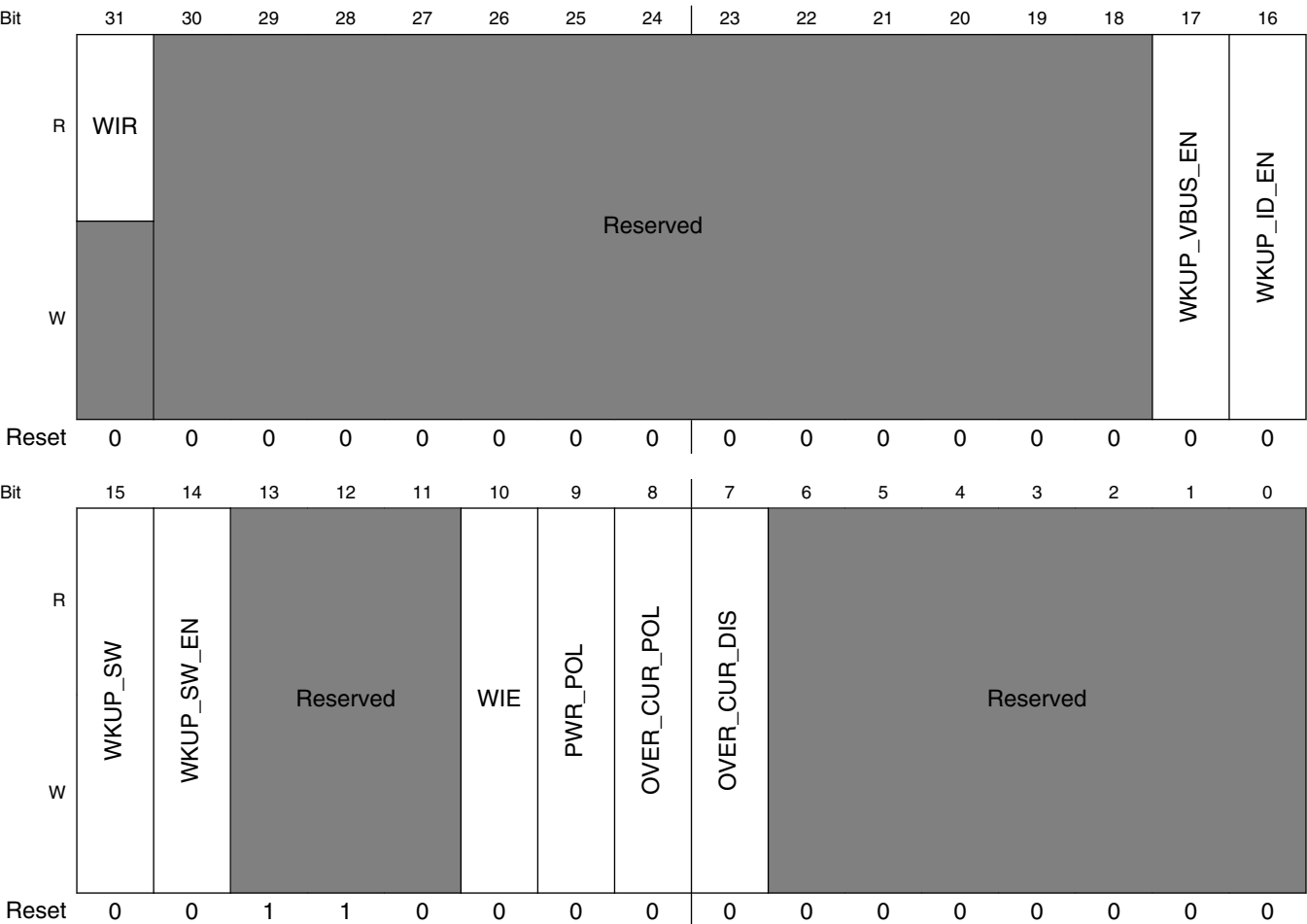
## USBNC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
218_4800	USB OTG1 Control Register (USBNC_USB_OTG1_CTRL)	32	R/W	0000_3000h	<a href="#">52.5.1/3182</a>
218_4804	USB OTG2 Control Register (USBNC_USB_OTG2_CTRL)	32	R/W	0000_3000h	<a href="#">52.5.2/3184</a>
218_4808	USB Host Control Register (USBNC_USB_UH_CTRL)	32	R/W	0000_1000h	<a href="#">52.5.3/3186</a>
218_4810	USB Host HSIC Control Register (USBNC_USB_UH_HSIC_CTRL)	32	R/W	0000_0042h	<a href="#">52.5.4/3188</a>
218_4818	OTG1 UTMI PHY Control 0 Register (USBNC_USB_OTG1_PHY_CTRL_0)	32	R/W	0000_0000h	<a href="#">52.5.5/3189</a>
218_481C	OTG2 UTMI PHY Control 0 Register (USBNC_USB_OTG2_PHY_CTRL_0)	32	R/W	0000_0098h	<a href="#">52.5.6/3190</a>

### 52.5.1 USB OTG1 Control Register (USBNC\_USB\_OTG1\_CTRL)

The USB OTG1 control register controls the integration specific features of the USB OTG1 module. These features are not directly related to the USB functionality, but control special features, interfacing on the USB ports, as well as power control and wake-up functionality.

Address: 218\_4000h base + 800h offset = 218\_4800h



USBNC\_USB\_OTG1\_CTRL field descriptions

Field	Description
31 WIR	<p>OTG1 Wake-up Interrupt Request</p> <p>This bit indicates that a wake-up interrupt request is received on the OTG1 port. This bit is cleared by disabling the wake-up interrupt (clearing bit "OWIE").</p> <p>1 Wake-up Interrupt Request received</p> <p>0 No wake-up interrupt request received</p>

Table continues on the next page...

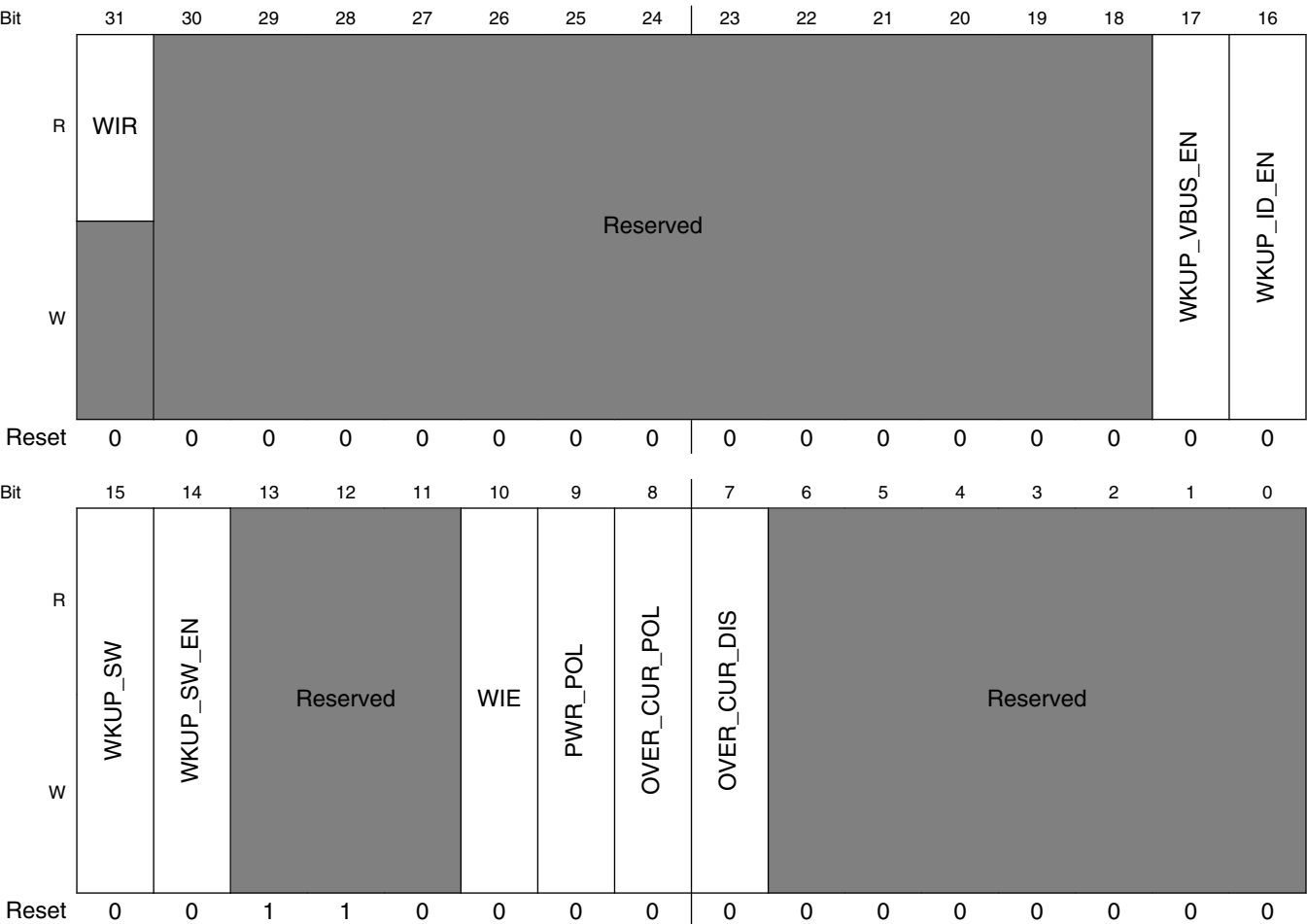
**USBNC\_USB\_OTG1\_CTRL field descriptions (continued)**

Field	Description
30–18 -	This field is reserved. Reserved
17 WKUP_VBUS_EN	OTG1 wake-up on VBUS change enable 1 Enable 0 Disable
16 WKUP_ID_EN	OTG1 Wake-up on ID change enable 1 Enable 0 Disable
15 WKUP_SW	OTG1 Software Wake-up 1 Force wake-up 0 Inactive
14 WKUP_SW_EN	OTG1 Software Wake-up Enable 1 Enable 0 Disable
13–11 -	This field is reserved. Reserved
10 WIE	OTG1 Wake-up Interrupt Enable This bit enables or disables the OTG1 wake-up interrupt. Disabling the interrupt also clears the Interrupt request bit. Wake-up interrupt enable should be turned off after receiving a wake-up interrupt and turned on again prior to going in suspend mode 1 Interrupt Enabled 0 Interrupt Disabled
9 PWR_POL	OTG1 Power Polarity This bit should be set according to PMIC Power Pin polarity. 1 PMIC Power Pin is High active. 0 PMIC Power Pin is Low active.
8 OVER_CUR_POL	OTG1 Polarity of Overcurrent The polarity of OTG1 port overcurrent event 1 Low active 0 High active
7 OVER_CUR_DIS	Disable OTG1 Overcurrent Detection 1 Disables overcurrent detection 0 Enables overcurrent detection
6–0 -	This field is reserved. Reserved

### 52.5.2 USB OTG2 Control Register (USBNC\_USB\_OTG2\_CTRL)

The USB OTG2 control register controls the integration specific features of the USB OTG2 module. These features are not directly related to the USB functionality, but control special features, interfacing on the USB ports, as well as power control and wake-up functionality.

Address: 218\_4000h base + 804h offset = 218\_4804h



USBNC\_USB\_OTG2\_CTRL field descriptions

Field	Description
31 WIR	<p>OTG2 Wake-up Interrupt Request</p> <p>This bit indicates that a wake-up interrupt request is received on the OTG port. This bit is cleared by disabling the wake-up interrupt (clearing bit "OWIE").</p> <p>1 Wake-up Interrupt Request received</p> <p>0 No wake-up interrupt request received</p>

Table continues on the next page...

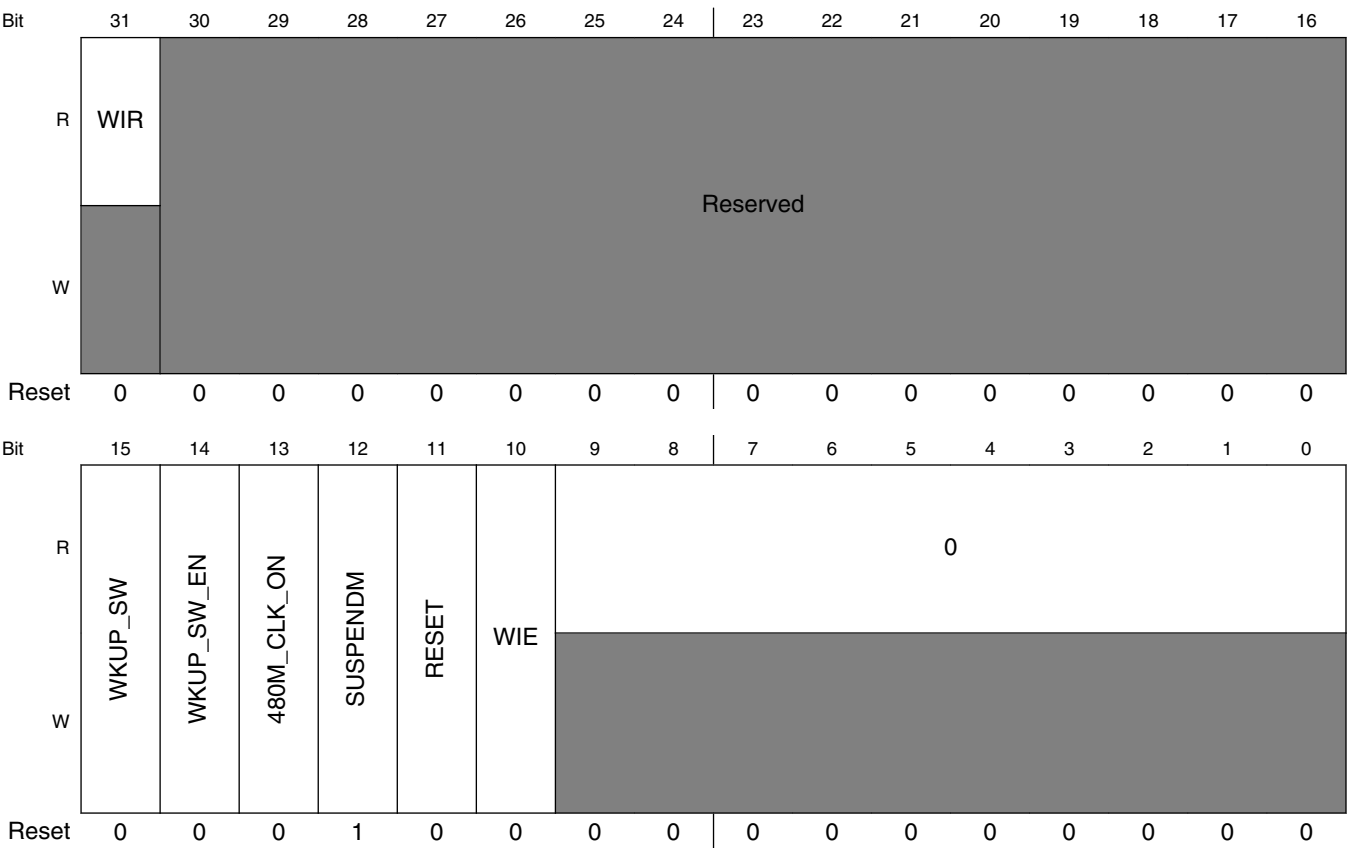
**USBNC\_USB\_OTG2\_CTRL field descriptions (continued)**

Field	Description
30–18 -	This field is reserved. Reserved
17 WKUP_VBUS_EN	OTG2 wake-up on VBUS change enable 1 Enable 0 Disable
16 WKUP_ID_EN	OTG2 Wake-up on ID change enable 1 Enable 0 Disable
15 WKUP_SW	OTG2 Software Wake-up 1 Force wake-up 0 Inactive
14 WKUP_SW_EN	OTG2 Software Wake-up Enable 1 Enable 0 Disable
13–11 -	This field is reserved. Reserved
10 WIE	OTG2 Wake-up Interrupt Enable This bit enables or disables the OTG2 wake-up interrupt. Disabling the interrupt also clears the Interrupt request bit. Wake-up interrupt enable should be turned off after receiving a wake-up interrupt and turned on again prior to going in suspend mode 1 Interrupt Enabled 0 Interrupt Disabled
9 PWR_POL	OTG2 Power Polarity This bit should be set according to PMIC Power Pin polarity. 1 PMIC Power Pin is High active. 0 PMIC Power Pin is Low active.
8 OVER_CUR_POL	OTG2 Polarity of Overcurrent The polarity of OTG2 port overcurrent event 1 Low active 0 High active
7 OVER_CUR_DIS	Disable OTG2 Overcurrent Detection 1 Disables overcurrent detection 0 Enables overcurrent detection
6–0 -	This field is reserved. Reserved

### 52.5.3 USB Host Control Register (USBNC\_USB\_UH\_CTRL)

The USB Host control register controls the integration specific features of the USB host module. These features are not directly related to the USB functionality, but control special features, interfacing on the USB ports, as well as power control and wake-up functionality.

Address: 218\_4000h base + 808h offset = 218\_4808h



USBNC\_USB\_UH\_CTRL field descriptions

Field	Description
31 WIR	Host Wake-up Interrupt Request This bit indicates that a wake-up interrupt request is received on the Host port. This bit is cleared by disabling the wake-up interrupt (clearing bit "WIE").  1 Wake-up Interrupt Request received 0 No wake-up interrupt request received
30-16 -	This field is reserved. Reserved

Table continues on the next page...

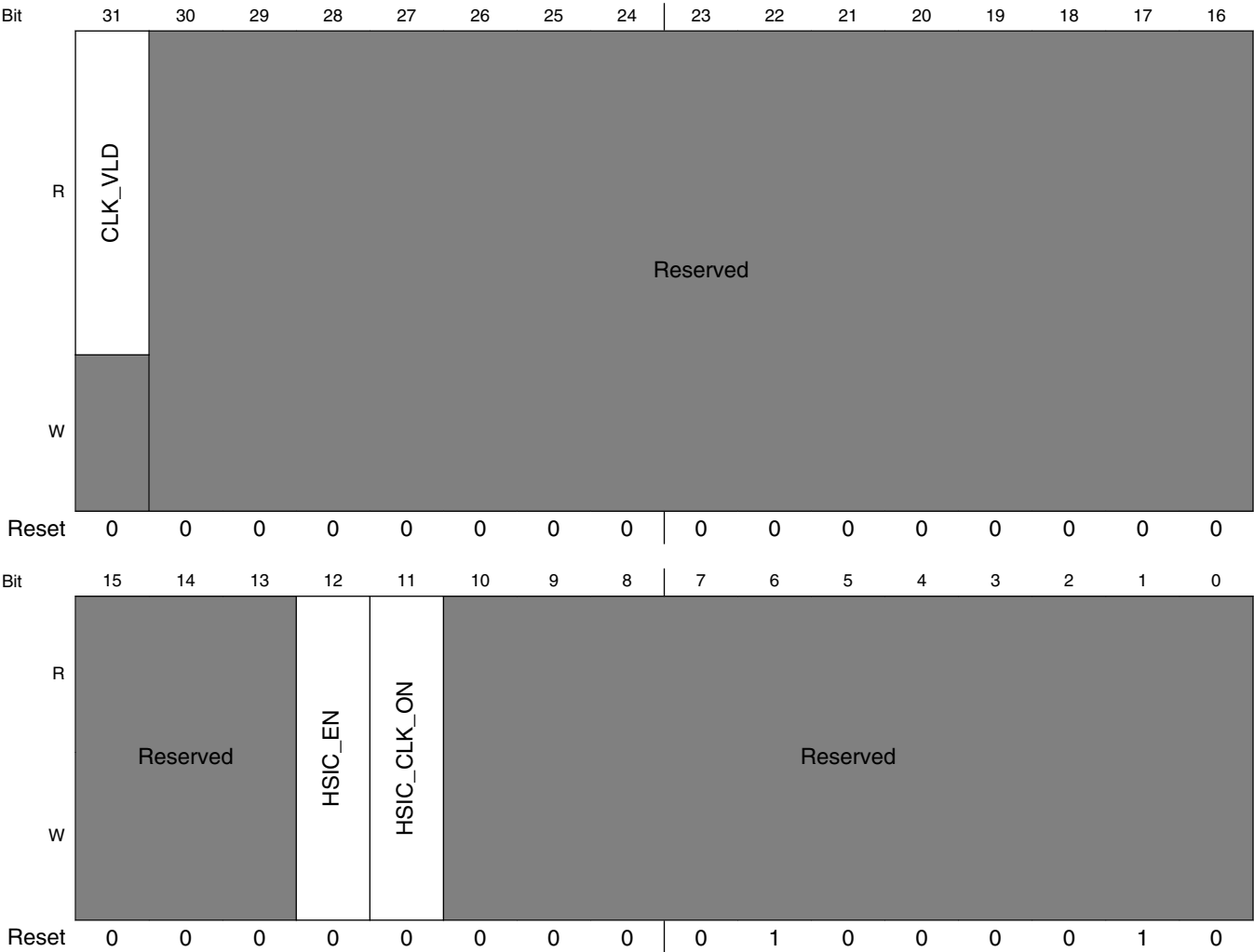
**USBNC\_USB\_UH\_CTRL field descriptions (continued)**

Field	Description
15 WKUP_SW	Host Software Wake-up 1 Force wake-up 0 Inactive
14 WKUP_SW_EN	Host Software Wake-up Enable 1 Enable 0 Disable
13 480M_CLK_ON	Force OTG UTMI PHY 480M clock output on when Host is not in suspend mode. 1 Force OTG UTMI PHY 480M clock output on 0 Inactive
12 SUSPENDM	Force Host UTMI PHY Suspend This bit is used to put PHY into suspend mode. During normal operation, S/W should set bits SUSP and PHCD in USB core register PORTSC1 to put PHY into suspend mode. For Freescale test only. 1 Disable 0 Enable
11 RESET	Force Host UTMI PHY Reset This bit is used to force a reset to the UTMI PHY. During normal operation, S/W should set USBCMD.RST bit to reset the UTMI PHY For Freescale test only. 1 Reset the PHY 0 Inactive
10 WIE	Host Wake-up Interrupt Enable This bit enables or disables the Host wake-up interrupt. Disabling the interrupt also clears the Interrupt request bit. Wake-up interrupt enable should be turned off after receiving a wake-up interrupt and turned on again prior to going in suspend mode 1 Interrupt Enabled 0 Interrupt Disabled
9–0 Reserved	This read-only field is reserved and always has the value 0.

### 52.5.4 USB Host HSIC Control Register (USBNC\_USB\_UH\_HSIC\_CTRL)

The USB Host HSIC control register controls Host high speed IC configuration. These features are not directly related to the USB functionality, but control special features, interfacing on the USB ports, as well as power control.

Address: 218\_4000h base + 810h offset = 218\_4810h



USBNC\_USB\_UH\_HSIC\_CTRL field descriptions

Field	Description
31 CLK_VLD	Indicating whether Host HSIC clock is valid.

Table continues on the next page...



**USBNC\_USB\_UH\_HSIC\_CTRL field descriptions (continued)**

Field	Description
	1 Valid 2 Invalid
30–13 -	This field is reserved. Reserved
12 HSIC_EN	Host HSIC enable  1 Enabled 0 Disabled
11 HSIC_CLK_ON	Force Host HSIC module 480M clock on, even when in Host is in suspend mode.  1 Active 0 Inactive
10–0 -	This field is reserved. Reserved

**52.5.5 OTG1 UTMI PHY Control 0 Register (USBNC\_USB\_OTG1\_PHY\_CTRL\_0)**

USB OTG1 UTMI PHY control register 0 is used to control the on-chip OTG1 UTMI PHY.

Address: 218\_4000h base + 818h offset = 218\_4818h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**USBNC\_USB\_OTG1\_PHY\_CTRL\_0 field descriptions**

Field	Description
31 UTMI_CLK_VLD	Indicating whether OTG1 UTMI PHY clock is valid

*Table continues on the next page...*

**USBNC\_USB\_OTG1\_PHY\_CTRL\_0 field descriptions (continued)**

Field	Description
	1 Valid 0 Invalid
30–3 -	This field is reserved. Reserved
2–0 -	This field is reserved. Reserved

**52.5.6 OTG2 UTMI PHY Control 0 Register (USBNC\_USB\_OTG2\_PHY\_CTRL\_0)**

USB OTG2 UTMI PHY Control Register 0 are used to control the on-chip OTG2 UTMI PHY.

Address: 218\_4000h base + 81Ch offset = 218\_481Ch

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	UTMI_CLK_VLD	Reserved														
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												Reserved			
W																
Reset	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0

**USBNC\_USB\_OTG2\_PHY\_CTRL\_0 field descriptions**

Field	Description
31 UTMI_CLK_VLD	Indicating whether OTG2 UTMI PHY clock is valid 1 Valid 0 Invalid
30–3 -	This field is reserved. Reserved
2–0 -	This field is reserved. Reserved

## 52.6 USB Core Memory Map/Register Definition

### USBC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
218_4000	Identification register (USBC_UOG1_ID)	32	R	E401_FA05h	<a href="#">52.6.1/3196</a>
218_4004	Hardware General (USBC_UOG1_HWGGENERAL)	32	R	0000_0015h	<a href="#">52.6.2/3197</a>
218_4008	Host Hardware Parameters (USBC_UOG1_HWHOST)	32	R	1002_0001h	<a href="#">52.6.3/3198</a>
218_400C	Device Hardware Parameters (USBC_UOG1_HWDEVICE)	32	R	0000_0011h	<a href="#">52.6.4/3199</a>
218_4010	TX Buffer Hardware Parameters (USBC_UOG1_HWTXBUF)	32	R	8008_0B08h	<a href="#">52.6.5/3199</a>
218_4014	RX Buffer Hardware Parameters (USBC_UOG1_HWRXBUF)	32	R	0000_0808h	<a href="#">52.6.6/3200</a>
218_4080	General Purpose Timer #0 Load (USBC_UOG1_GPTIMER0LD)	32	R/W	0000_0000h	<a href="#">52.6.7/3201</a>
218_4084	General Purpose Timer #0 Controller (USBC_UOG1_GPTIMER0CTRL)	32	R/W	0000_0000h	<a href="#">52.6.8/3201</a>
218_4088	General Purpose Timer #1 Load (USBC_UOG1_GPTIMER1LD)	32	R/W	0000_0000h	<a href="#">52.6.9/3203</a>
218_408C	General Purpose Timer #1 Controller (USBC_UOG1_GPTIMER1CTRL)	32	R/W	0000_0000h	<a href="#">52.6.10/3203</a>
218_4090	System Bus Config (USBC_UOG1_SBUSCFG)	32	R/W	0000_0002h	<a href="#">52.6.11/3204</a>
218_4100	Capability Registers Length (USBC_UOG1_CAPLENGTH)	8	R	40h	<a href="#">52.6.12/3205</a>
218_4102	Host Controller Interface Version (USBC_UOG1_HCIVERSION)	16	R	0100h	<a href="#">52.6.13/3206</a>
218_4104	Host Controller Structural Parameters (USBC_UOG1_HCSPARAMS)	32	R	0001_0011h	<a href="#">52.6.14/3206</a>
218_4108	Host Controller Capability Parameters (USBC_UOG1_HCCPARAMS)	32	R	0000_0006h	<a href="#">52.6.15/3208</a>
218_4120	Device Controller Interface Version (USBC_UOG1_DCIVERSION)	16	R	0001h	<a href="#">52.6.16/3210</a>
218_4124	Device Controller Capability Parameters (USBC_UOG1_DCCPARAMS)	32	R	0000_0188h	<a href="#">52.6.17/3210</a>
218_4140	USB Command Register (USBC_UOG1_USBCMD)	32	R/W	0008_0000h	<a href="#">52.6.18/3212</a>
218_4144	USB Status Register (USBC_UOG1_USBSTS)	32	R/W	0000_0000h	<a href="#">52.6.19/3216</a>
218_4148	Interrupt Enable Register (USBC_UOG1_USBINTR)	32	R/W	0000_0000h	<a href="#">52.6.20/3220</a>
218_414C	USB Frame Index (USBC_UOG1_FRINDEX)	32	R/W	0000_0000h	<a href="#">52.6.21/3222</a>
218_4154	Frame List Base Address (USBC_UOG1_PERIODICLISTBASE)	32	R/W	0000_0000h	<a href="#">52.6.22/3223</a>

Table continues on the next page...

## USBC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
218_4154	Device Address (USBC_UOG1_DEVICEADDR)	32	R/W	0000_0000h	<a href="#">52.6.23/3223</a>
218_4158	Next Asynch. Address (USBC_UOG1_ASYNCCLISTADDR)	32	R/W	0000_0000h	<a href="#">52.6.24/3224</a>
218_4158	Endpoint List Address (USBC_UOG1_ENDPTLISTADDR)	32	R/W	0000_0000h	<a href="#">52.6.25/3225</a>
218_4160	Programmable Burst Size (USBC_UOG1_BURSTSIZE)	32	R/W	0000_0000h	<a href="#">52.6.26/3226</a>
218_4164	TX FIFO Fill Tuning (USBC_UOG1_TXFILLTUNING)	32	R/W	0000_0808h	<a href="#">52.6.27/3226</a>
218_4178	Endpoint NAK (USBC_UOG1_ENDPTNAK)	32	R/W	0000_0000h	<a href="#">52.6.28/3228</a>
218_417C	Endpoint NAK Enable (USBC_UOG1_ENDPTNAKEN)	32	R/W	0000_0000h	<a href="#">52.6.29/3228</a>
218_4180	Configure Flag Register (USBC_UOG1_CONFIGFLAG)	32	R/W	0000_0001h	<a href="#">52.6.30/3229</a>
218_4184	Port Status & Control (USBC_UOG1_PORTSC1)	32	R/W	1000_0000h	<a href="#">52.6.31/3230</a>
218_41A4	On-The-Go Status & control (USBC_UOG1_OTGSC)	32	R/W	0000_0120h	<a href="#">52.6.32/3236</a>
218_41A8	USB Device Mode (USBC_UOG1_USBMODE)	32	R/W	0000_0000h	<a href="#">52.6.33/3240</a>
218_41AC	Endpoint Setup Status (USBC_UOG1_ENDPTSETUPSTAT)	32	R/W	0000_0000h	<a href="#">52.6.34/3241</a>
218_41B0	Endpoint Prime (USBC_UOG1_ENDPTPRIME)	32	R/W	0000_0000h	<a href="#">52.6.35/3242</a>
218_41B4	Endpoint Flush (USBC_UOG1_ENDPTFLUSH)	32	R/W	0000_0000h	<a href="#">52.6.36/3243</a>
218_41B8	Endpoint Status (USBC_UOG1_ENDPTSTAT)	32	R	0000_0000h	<a href="#">52.6.37/3243</a>
218_41BC	Endpoint Complete (USBC_UOG1_ENDPTCOMPLETE)	32	R/W	0000_0000h	<a href="#">52.6.38/3244</a>
218_41C0	Endpoint Control0 (USBC_UOG1_ENDPTCTRL0)	32	R/W	0080_0080h	<a href="#">52.6.39/3245</a>
218_41C4	Endpoint Control 1 (USBC_UOG1_ENDPTCTRL1)	32	R/W	0000_0000h	<a href="#">52.6.40/3246</a>
218_41C8	Endpoint Control 2 (USBC_UOG1_ENDPTCTRL2)	32	R/W	0000_0000h	<a href="#">52.6.41/3249</a>
218_41CC	Endpoint Control 3 (USBC_UOG1_ENDPTCTRL3)	32	R/W	0000_0000h	<a href="#">52.6.42/3252</a>
218_41D0	Endpoint Control 4 (USBC_UOG1_ENDPTCTRL4)	32	R/W	0000_0000h	<a href="#">52.6.43/3254</a>
218_41D4	Endpoint Control 5 (USBC_UOG1_ENDPTCTRL5)	32	R/W	0000_0000h	<a href="#">52.6.44/3257</a>

Table continues on the next page...

## USBC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
218_41D8	Endpoint Control 6 (USBC_UOG1_ENDPTCTRL6)	32	R/W	0000_0000h	<a href="#">52.6.45/3260</a>
218_41DC	Endpoint Control 7 (USBC_UOG1_ENDPTCTRL7)	32	R/W	0000_0000h	<a href="#">52.6.46/3262</a>
218_4200	Identification register (USBC_UOG2_ID)	32	R	E401_FA05h	<a href="#">52.6.1/3196</a>
218_4204	Hardware General (USBC_UOG2_HWGENERAL)	32	R	0000_0015h	<a href="#">52.6.2/3197</a>
218_4208	Host Hardware Parameters (USBC_UOG2_HWHOST)	32	R	1002_0001h	<a href="#">52.6.3/3198</a>
218_420C	Device Hardware Parameters (USBC_UOG2_HWDEVICE)	32	R	0000_0011h	<a href="#">52.6.4/3199</a>
218_4210	TX Buffer Hardware Parameters (USBC_UOG2_HWTXBUF)	32	R	8008_0B08h	<a href="#">52.6.5/3199</a>
218_4214	RX Buffer Hardware Parameters (USBC_UOG2_HWRXBUF)	32	R	0000_0808h	<a href="#">52.6.6/3200</a>
218_4280	General Purpose Timer #0 Load (USBC_UOG2_GPTIMER0LD)	32	R/W	0000_0000h	<a href="#">52.6.7/3201</a>
218_4284	General Purpose Timer #0 Controller (USBC_UOG2_GPTIMER0CTRL)	32	R/W	0000_0000h	<a href="#">52.6.8/3201</a>
218_4288	General Purpose Timer #1 Load (USBC_UOG2_GPTIMER1LD)	32	R/W	0000_0000h	<a href="#">52.6.9/3203</a>
218_428C	General Purpose Timer #1 Controller (USBC_UOG2_GPTIMER1CTRL)	32	R/W	0000_0000h	<a href="#">52.6.10/3203</a>
218_4290	System Bus Config (USBC_UOG2_SBUSCFG)	32	R/W	0000_0002h	<a href="#">52.6.11/3204</a>
218_4300	Capability Registers Length (USBC_UOG2_CAPLENGTH)	8	R	40h	<a href="#">52.6.12/3205</a>
218_4302	Host Controller Interface Version (USBC_UOG2_HCIVERSION)	16	R	0100h	<a href="#">52.6.13/3206</a>
218_4304	Host Controller Structural Parameters (USBC_UOG2_HCSPARAMS)	32	R	0001_0011h	<a href="#">52.6.14/3206</a>
218_4308	Host Controller Capability Parameters (USBC_UOG2_HCCPARAMS)	32	R	0000_0006h	<a href="#">52.6.15/3208</a>
218_4320	Device Controller Interface Version (USBC_UOG2_DCIVERSION)	16	R	0001h	<a href="#">52.6.16/3210</a>
218_4324	Device Controller Capability Parameters (USBC_UOG2_DCCPARAMS)	32	R	0000_0188h	<a href="#">52.6.17/3210</a>
218_4340	USB Command Register (USBC_UOG2_USBCMD)	32	R/W	0008_0000h	<a href="#">52.6.18/3212</a>
218_4344	USB Status Register (USBC_UOG2_USBSTS)	32	R/W	0000_0000h	<a href="#">52.6.19/3216</a>
218_4348	Interrupt Enable Register (USBC_UOG2_USBINTR)	32	R/W	0000_0000h	<a href="#">52.6.20/3220</a>
218_434C	USB Frame Index (USBC_UOG2_FRINDEX)	32	R/W	0000_0000h	<a href="#">52.6.21/3222</a>
218_4354	Frame List Base Address (USBC_UOG2_PERIODICLISTBASE)	32	R/W	0000_0000h	<a href="#">52.6.22/3223</a>

Table continues on the next page...

## USBC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
218_4354	Device Address (USBC_UOG2_DEVICEADDR)	32	R/W	0000_0000h	<a href="#">52.6.23/3223</a>
218_4358	Next Asynch. Address (USBC_UOG2_ASYNCLISTADDR)	32	R/W	0000_0000h	<a href="#">52.6.24/3224</a>
218_4358	Endpoint List Address (USBC_UOG2_ENDPTLISTADDR)	32	R/W	0000_0000h	<a href="#">52.6.25/3225</a>
218_4360	Programmable Burst Size (USBC_UOG2_BURSTSIZE)	32	R/W	0000_0000h	<a href="#">52.6.26/3226</a>
218_4364	TX FIFO Fill Tuning (USBC_UOG2_TXFILLTUNING)	32	R/W	0000_0808h	<a href="#">52.6.27/3226</a>
218_4378	Endpoint NAK (USBC_UOG2_ENDPTNAK)	32	R/W	0000_0000h	<a href="#">52.6.28/3228</a>
218_437C	Endpoint NAK Enable (USBC_UOG2_ENDPTNAKEN)	32	R/W	0000_0000h	<a href="#">52.6.29/3228</a>
218_4380	Configure Flag Register (USBC_UOG2_CONFIGFLAG)	32	R/W	0000_0001h	<a href="#">52.6.30/3229</a>
218_4384	Port Status & Control (USBC_UOG2_PORTSC1)	32	R/W	1000_0000h	<a href="#">52.6.31/3230</a>
218_43A4	On-The-Go Status & control (USBC_UOG2_OTGSC)	32	R/W	0000_0120h	<a href="#">52.6.32/3236</a>
218_43A8	USB Device Mode (USBC_UOG2_USBMODE)	32	R/W	0000_0000h	<a href="#">52.6.33/3240</a>
218_43AC	Endpoint Setup Status (USBC_UOG2_ENDPTSETUPSTAT)	32	R/W	0000_0000h	<a href="#">52.6.34/3241</a>
218_43B0	Endpoint Prime (USBC_UOG2_ENDPTPRIME)	32	R/W	0000_0000h	<a href="#">52.6.35/3242</a>
218_43B4	Endpoint Flush (USBC_UOG2_ENDPTFLUSH)	32	R/W	0000_0000h	<a href="#">52.6.36/3243</a>
218_43B8	Endpoint Status (USBC_UOG2_ENDPTSTAT)	32	R	0000_0000h	<a href="#">52.6.37/3243</a>
218_43BC	Endpoint Complete (USBC_UOG2_ENDPTCOMPLETE)	32	R/W	0000_0000h	<a href="#">52.6.38/3244</a>
218_43C0	Endpoint Control0 (USBC_UOG2_ENDPTCTRL0)	32	R/W	0080_0080h	<a href="#">52.6.39/3245</a>
218_43C4	Endpoint Control 1 (USBC_UOG2_ENDPTCTRL1)	32	R/W	0000_0000h	<a href="#">52.6.40/3246</a>
218_43C8	Endpoint Control 2 (USBC_UOG2_ENDPTCTRL2)	32	R/W	0000_0000h	<a href="#">52.6.41/3249</a>
218_43CC	Endpoint Control 3 (USBC_UOG2_ENDPTCTRL3)	32	R/W	0000_0000h	<a href="#">52.6.42/3252</a>
218_43D0	Endpoint Control 4 (USBC_UOG2_ENDPTCTRL4)	32	R/W	0000_0000h	<a href="#">52.6.43/3254</a>
218_43D4	Endpoint Control 5 (USBC_UOG2_ENDPTCTRL5)	32	R/W	0000_0000h	<a href="#">52.6.44/3257</a>

Table continues on the next page...

## USBC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
218_43D8	Endpoint Control 6 (USBC_UOG2_ENDPTCTRL6)	32	R/W	0000_0000h	<a href="#">52.6.45/3260</a>
218_43DC	Endpoint Control 7 (USBC_UOG2_ENDPTCTRL7)	32	R/W	0000_0000h	<a href="#">52.6.46/3262</a>
218_4400	Identification register (USBC_UH1_ID)	32	R	E401_FA05h	<a href="#">52.6.1/3196</a>
218_4404	Hardware General (USBC_UH1_HWGGENERAL)	32	R	0000_0015h	<a href="#">52.6.2/3197</a>
218_4408	Host Hardware Parameters (USBC_UH1_HWHOST)	32	R	1002_0001h	<a href="#">52.6.3/3198</a>
218_4410	TX Buffer Hardware Parameters (USBC_UH1_HWTXBUF)	32	R	8008_0B08h	<a href="#">52.6.5/3199</a>
218_4414	RX Buffer Hardware Parameters (USBC_UH1_HWRXBUF)	32	R	0000_0808h	<a href="#">52.6.6/3200</a>
218_4480	General Purpose Timer #0 Load (USBC_UH1_GPTIMER0LD)	32	R/W	0000_0000h	<a href="#">52.6.7/3201</a>
218_4484	General Purpose Timer #0 Controller (USBC_UH1_GPTIMER0CTRL)	32	R/W	0000_0000h	<a href="#">52.6.8/3201</a>
218_4488	General Purpose Timer #1 Load (USBC_UH1_GPTIMER1LD)	32	R/W	0000_0000h	<a href="#">52.6.9/3203</a>
218_448C	General Purpose Timer #1 Controller (USBC_UH1_GPTIMER1CTRL)	32	R/W	0000_0000h	<a href="#">52.6.10/3203</a>
218_4490	System Bus Config (USBC_UH1_SBUSCFG)	32	R/W	0000_0002h	<a href="#">52.6.11/3204</a>
218_4500	Capability Registers Length (USBC_UH1_CAPLENGTH)	8	R	40h	<a href="#">52.6.12/3205</a>
218_4502	Host Controller Interface Version (USBC_UH1_HCIVERSION)	16	R	0100h	<a href="#">52.6.13/3206</a>
218_4504	Host Controller Structural Parameters (USBC_UH1_HCSPARAMS)	32	R	0001_0011h	<a href="#">52.6.14/3206</a>
218_4508	Host Controller Capability Parameters (USBC_UH1_HCCPARAMS)	32	R	0000_0006h	<a href="#">52.6.15/3208</a>
218_4540	USB Command Register (USBC_UH1_USBCMD)	32	R/W	0008_0000h	<a href="#">52.6.18/3212</a>
218_4544	USB Status Register (USBC_UH1_USBSTS)	32	R/W	0000_0000h	<a href="#">52.6.19/3216</a>
218_4548	Interrupt Enable Register (USBC_UH1_USBINTR)	32	R/W	0000_0000h	<a href="#">52.6.20/3220</a>
218_454C	USB Frame Index (USBC_UH1_FRINDEX)	32	R/W	0000_0000h	<a href="#">52.6.21/3222</a>
218_4554	Frame List Base Address (USBC_UH1_PERIODICLISTBASE)	32	R/W	0000_0000h	<a href="#">52.6.22/3223</a>
218_4558	Next Asynch. Address (USBC_UH1_ASYNCCLISTADDR)	32	R/W	0000_0000h	<a href="#">52.6.24/3224</a>
218_4560	Programmable Burst Size (USBC_UH1_BURSTSIZE)	32	R/W	0000_0000h	<a href="#">52.6.26/3226</a>
218_4564	TX FIFO Fill Tuning (USBC_UH1_TXFILLTUNING)	32	R/W	0000_0808h	<a href="#">52.6.27/3226</a>

Table continues on the next page...

## USBC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
218_4580	Configure Flag Register (USBC_UH1_CONFIGFLAG)	32	R/W	0000_0001h	<a href="#">52.6.30/3229</a>
218_4584	Port Status & Control (USBC_UH1_PORTSC1)	32	R/W	1000_0000h	<a href="#">52.6.31/3230</a>
218_45A8	USB Device Mode (USBC_UH1_USBMODE)	32	R/W	0000_0000h	<a href="#">52.6.33/3240</a>

## 52.6.1 Identification register (USBC\_n\_ID)

The ID register identifies the USB 2.0 High-Speed core and its revision.

Address: 218\_4000h base + 0h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								REVISION							
W																
Reset	1	1	1	0	0	1	0	0	0	0	0	0	0	0	0	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved			NID					Reserved		ID					
W																
Reset	1	1	1	1	1	0	1	0	0	0	0	0	0	1	0	1

## USBC\_n\_ID field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 REVISION	Revision number of the controller core.
15–14 -	This field is reserved. Reserved
13–8 NID	Complement version of ID
7–6 -	This field is reserved. Reserved
5–0 ID	Configuration number. This number is set to 0x05 and indicates that the peripheral is USB 2.0 High-Speed core.



## 52.6.2 Hardware General (USBC\_n\_HWGENERAL)

General hardware parameters as defined in System Level Issues and Core Configuration.

### NOTE

The reset value could vary from instance to instance. Please see the detail in bit field description and ignore reset value in summary table in this case!

Address: 218\_4000h base + 4h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved					SM		PHYM			PHYW		Reserved			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	1

### USBC\_n\_HWGENERAL field descriptions

Field	Description
31–11 -	This field is reserved. Reserved
10–9 SM	Serial interface mode capability SM bit reset value is '00b'  00 No Serial Engine, always use parallel signalling. 01 Serial Engine present, always use serial signalling for FS/LS. 10 Software programmable - Reset to use parallel signalling for FS/LS 11 Software programmable - Reset to use serial signalling for FS/LS
8–6 PHYM	Transciever type PHYM bit reset value: '0000b' for OTG controller core, '0100b' for Host-only controller core.  000 UTMI/UMTI+ 001 ULPI DDR 010 ULPI 011 Serial Only 100 Software programmable - reset to UTMI/UTMI+ 101 Software programmable - reset to ULPI DDR 110 Software programmable - reset to ULPI 111 Software programmable - reset to Serial 1000 IC-USB 1001 Software programmable - reset to IC-USB

Table continues on the next page...

**USBC\_n\_HWGENERAL field descriptions (continued)**

Field	Description
1010 HSIC	
1011 Software programmable - reset to HSIC	
5–4 PHYW	Data width of the transceiver connected to the controller core. PHYW bit reset value is '01b'.  00 8 bit wide data bus Software non-programmable 01 16 bit wide data bus Software non-programmable 10 Reset to 8 bit wide data bus Software programmable 11 Reset to 16 bit wide data bus Software programmable
3–0 -	This field is reserved. Reserved

**52.6.3 Host Hardware Parameters (USBC\_n\_HWHOST)**

Address: 218\_4000h base + 8h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												NPORT		HC	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

**USBC\_n\_HWHOST field descriptions**

Field	Description
31–4 -	This field is reserved. Reserved
3–1 NPORT	The Number of downstream ports supported by the host controller is (NPORT+1). <b>NOTE:</b> When these bits value is '000', it indicates a single-port host controller.
0 HC	Host Capable. Indicating whether host operation mode is supported or not.  1 Supported 0 Not supported

## 52.6.4 Device Hardware Parameters (USBC\_n\_HWDEVICE)

### NOTE

This register is only available in OTG core.

Address: 218\_4000h base + Ch offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved										DEVEP				DC	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

### USBC\_n\_HWDEVICE field descriptions

Field	Description
31–6 -	This field is reserved. Reserved
5–1 DEVEP	Device Endpoint Number
0 DC	Device Capable. Indicating whether device operation mode is supported or not.  1 Supported 0 Not supported

## 52.6.5 TX Buffer Hardware Parameters (USBC\_n\_HWTXBUF)

Address: 218\_4000h base + 10h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								TXCHANADD								Reserved								TXBURST							
W																																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	1	1	0	0	0	0	1	0	0	0

**USBC\_n\_HWTXBUF field descriptions**

Field	Description
31–24 -	This field is reserved. Reserved
23–16 TXCHANADD	TX FIFO Buffer size is: $(2^{\text{TXCHANADD}}) * 4$ Bytes. These bits are set to '08h', so buffer size is 256*4 Bytes. For the OTG controller operating in device mode, this is the FIFO buffer size per endpoint. As the OTG controller has 8 TX endpoint, there are 8 of these buffers. For the OTG controller operating in host mode, or for Host-only controller, the entire buffer memory is used as a single TX buffer. Therefore, there is only 1 of this buffer
15–8 -	This field is reserved. Reserved
7–0 TXBURST	Default burst size for memory to TX buffer transfer. This is reset value of TXPBURST bits in USB core regster USB_n_BURSTSIZE. Please see <a href="#">Programmable Burst Size (USBC_n_BURSTSIZE)</a> .

**52.6.6 RX Buffer Hardware Parameters (USBC\_n\_HWRXBUF)**

Address: 218\_4000h base + 14h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																RXADD				RXBURST											
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0

**USBC\_n\_HWRXBUF field descriptions**

Field	Description
31–16 -	This field is reserved. Reserved
15–8 RXADD	Buffer total size for all receive endpoints is $(2^{\text{RXADD}})$ . RX Buffer size is: $(2^{\text{RXADD}}) * 4$ Bytes. These bits are set to '08h', so buffer size is 256*4 Bytes. There is a single Receive FIFO buffer in the USB controller. The buffer is shared for all endpoints for the OTG controller in device mode.
7–0 RXBURST	Default burst size for memory to RX buffer transfer. This is reset value of RXPBURST bits in USB core regster USB_n_BURSTSIZE. Please see <a href="#">Programmable Burst Size (USBC_n_BURSTSIZE)</a> .

## 52.6.7 General Purpose Timer #0 Load (USBC\_n\_GPTIMER0LD)

This register controls load value of the count timer in register n\_GPTIMER0CTRL. Please see [General Purpose Timer #0 Controller \(USBC\\_n\\_GPTIMER0CTRL\)](#) .

Address: 218\_4000h base + 80h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
R																																								
W	Reserved																								GPTLD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0								

### USBC\_n\_GPTIMER0LD field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–0 GPTLD	General Purpose Timer Load Value These bit fields are loaded to GPTCNT bits when GPTRST bit is set '1b'. This value represents the time in microseconds minus 1 for the timer duration. Example: for a one millisecond timer, load 1000-1=999 or 0x0003E7. <b>NOTE:</b> Max value is 0xFFFFF or 16.777215 seconds.

## 52.6.8 General Purpose Timer #0 Controller (USBC\_n\_GPTIMER0CTRL)

This register contains the control for this countdown timer and a data field can be queried to determine the running count value. This timer has granularity on 1 us and can be programmed to a little over 16 seconds. There are two counter modes which are described in the register table below. When the timer counter value transitions to zero, an interrupt could be generated if enable.

Interrupt status bit is TI0 bit in n\_USBSTS register (See [USB Status Register \(USBC\\_n\\_USBSTS\)](#) ), interrupt enable bit is TIE0 bit in n\_USBINTR register. (See [Interrupt Enable Register \(USBC\\_n\\_USBINTR\)](#) .)

## USB Core Memory Map/Register Definition

Address: 218\_4000h base + 84h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GPTRUN	GPTRST	Reserved						GPTMODE	GPTCNT						
W	GPTRUN	GPTRST							GPTMODE							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPTCNT															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### USBC\_n\_GPTIMER0CTRL field descriptions

Field	Description
31 GPTRUN	General Purpose Timer Run GPTCNT bits are not effected when setting or clearing this bit.  0 Stop counting 1 Run
30 GPTRST	General Purpose Timer Reset  0 No action 1 Load counter value from GPTLD bits in n_GPTIMEROLD
29–25 -	This field is reserved. Reserved
24 GPTMODE	General Purpose Timer Mode  In one shot mode, the timer will count down to zero, generate an interrupt, and stop until the counter is reset by software;  In repeat mode, the timer will count down to zero, generate an interrupt and automatically reload the counter value from GPTLD bits to start again.  0 One Shot Mode 1 Repeat Mode
23–0 GPTCNT	General Purpose Timer Counter. This field is the count value of the countdown timer.

## 52.6.9 General Purpose Timer #1 Load (USBC\_n\_GPTIMER1LD)

This register controls load value of the count timer in register n\_GPTIMER1CTRL. Please see [General Purpose Timer #1 Controller \(USBC\\_n\\_GPTIMER1CTRL\)](#).

Address: 218\_4000h base + 88h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0								
R																																								
W	Reserved																								GPTLD															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0								

### USBC\_n\_GPTIMER1LD field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–0 GPTLD	General Purpose Timer Load Value These bit fields are loaded to GPTCNT bits when GPTRST bit is set '1b'. This value represents the time in microseconds minus 1 for the timer duration. Example: for a one millisecond timer, load 1000-1=999 or 0x0003E7. <b>NOTE:</b> Max value is 0xFFFFF or 16.777215 seconds.

## 52.6.10 General Purpose Timer #1 Controller (USBC\_n\_GPTIMER1CTRL)

This register contains the control for this countdown timer and a data field can be queried to determine the running count value. This timer has granularity on 1 us and can be programmed to a little over 16 seconds. There are two counter modes which are described in the register table below. When the timer counter value transitions to zero, an interrupt could be generated if enable.

Interrupt status bit is TI1 bit in USB\_n\_USBSTS register (See [USB Status Register \(USBC\\_n\\_USBSTS\)](#)), interrupt enable bit is TIE1 bit in n\_USBINTR register (See [Interrupt Enable Register \(USBC\\_n\\_USBINTR\)](#)).

## USB Core Memory Map/Register Definition

Address: 218\_4000h base + 8Ch offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	GPTRUN	GPTRST	Reserved						GPTMODE	GPTCNT						
W	GPTRUN	GPTRST	Reserved						GPTMODE	GPTCNT						
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	GPTCNT															
W	GPTCNT															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### USBC\_n\_GPTIMER1CTRL field descriptions

Field	Description
31 GPTRUN	General Purpose Timer Run GPTCNT bits are not effected when setting or clearing this bit.  0 Stop counting 1 Run
30 GPTRST	General Purpose Timer Reset  0 No action 1 Load counter value from GPTLD bits in USB_n_GPTIMER0LD
29–25 -	This field is reserved. Reserved
24 GPTMODE	General Purpose Timer Mode  In one shot mode, the timer will count down to zero, generate an interrupt, and stop until the counter is reset by software. In repeat mode, the timer will count down to zero, generate an interrupt and automatically reload the counter value from GPTLD bits to start again.  0 One Shot Mode 1 Repeat Mode
23–0 GPTCNT	General Purpose Timer Counter.  This field is the count value of the countdown timer.

## 52.6.11 System Bus Config (USBC\_n\_SBUSCFG)

Address: 218\_4000h base + 90h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																AHBBRS															
W	Reserved																T															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0



**USBC\_n\_SBUSCFG field descriptions**

Field	Description
31–3 -	This field is reserved. Reserved
2–0 AHBBRST	<p>AHB master interface Burst configuration</p> <p>These bits control AHB master transfer type sequence (or priority).</p> <p><b>NOTE:</b> This register overrides n_BURSTSIZE register when its value is not zero.</p> <p>000 Incremental burst of unspecified length only</p> <p>001 INCR4 burst, then single transfer</p> <p>010 INCR8 burst, INCR4 burst, then single transfer</p> <p>011 INCR16 burst, INCR8 burst, INCR4 burst, then single transfer</p> <p>100 Reserved, don't use</p> <p>101 INCR4 burst, then incremental burst of unspecified length</p> <p>110 INCR8 burst, INCR4 burst, then incremental burst of unspecified length</p> <p>111 INCR16 burst, INCR8 burst, INCR4 burst, then incremental burst of unspecified length</p>

**52.6.12 Capability Registers Length (USBC\_n\_CAPLENGTH)**

The Capability Registers Length register contains the address offset to the Operational registers relative to the CAPLENGTH register.

Address: 218\_4000h base + 100h offset + (512d × i), where i=0d to 2d

Bit	7	6	5	4	3	2	1	0
Read	CAPLENGTH							
Write								
Reset	0	1	0	0	0	0	0	0

**USBC\_n\_CAPLENGTH field descriptions**

Field	Description
7–0 CAPLENGTH	<p>These bits are used as an offset to add to register base to find the beginning of the Operational Register.</p> <p>Default value is '40h'.</p>

### 52.6.13 Host Controller Interface Version (USBC\_n\_HCIVERSION)

This is a 2-byte register containing a BCD encoding of the EHCI revision number supported by this host controller. The most significant byte of this register represents a major revision and the least significant byte is the minor revision.

Address: 218\_4000h base + 102h offset + (512d × i), where i=0d to 2d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	HCIVERSION															
Write																
Reset	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0

USBC\_n\_HCIVERSION field descriptions

Field	Description
15–0 HCIVERSION	Host Controller Interface Version Number Default value is '10h', which means EHCI rev1.0.

### 52.6.14 Host Controller Structural Parameters (USBC\_n\_HCSPARAMS)

The following figure shows the port steering logic capabilities of Host Control Structural Parameters (n\_HCSPARAMS).

Address: 218\_4000h base + 104h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	Reserved				N_TT				N_PTT				Reserved				PI
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	N_CC				N_PCC				Reserved				PPC	N_PORTS		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	1

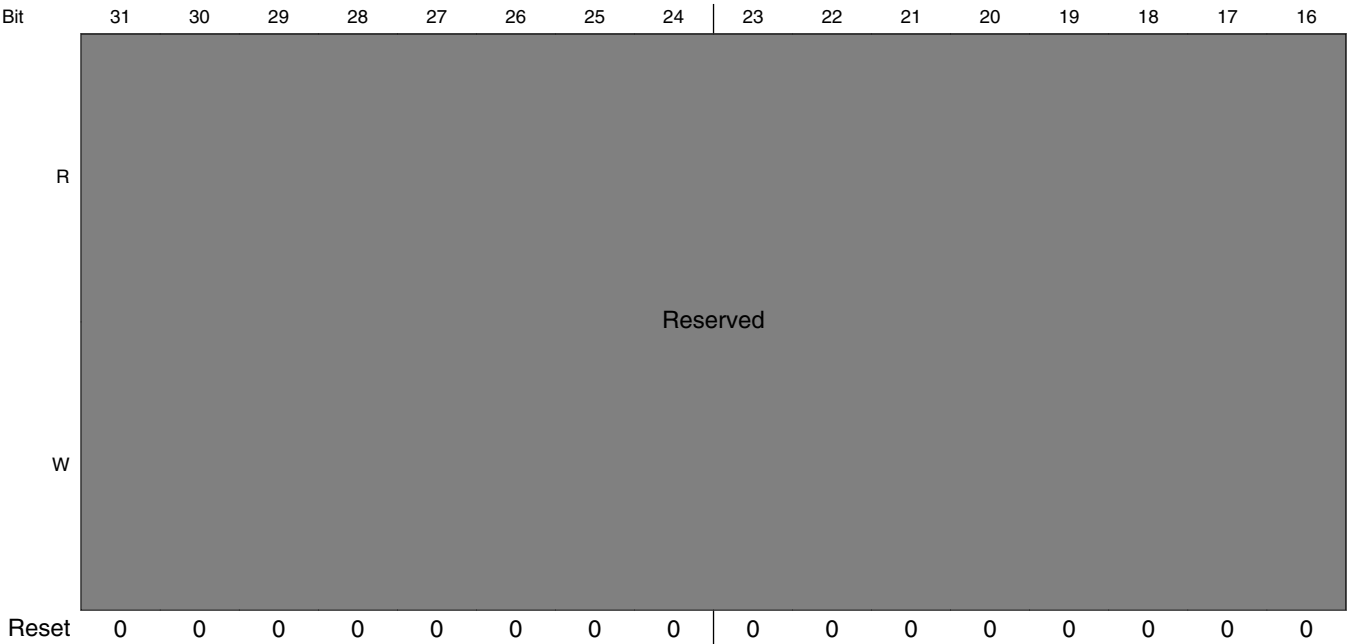
**USBC\_n\_HCSPARAMS field descriptions**

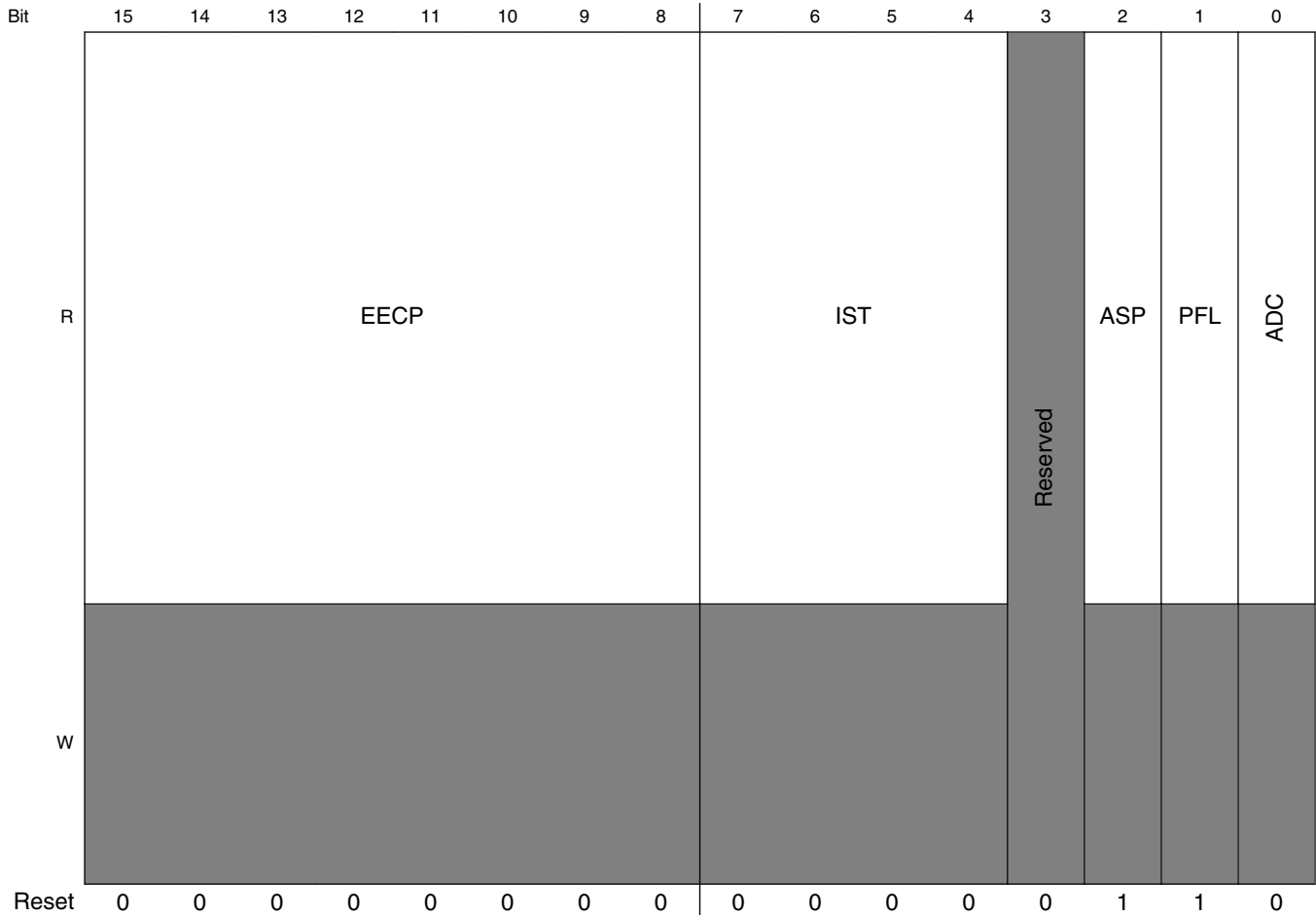
Field	Description
31–28 -	This field is reserved. Reserved
27–24 N_TT	Number of Transaction Translators (N_TT). Default value '0000b'  This field indicates the number of embedded transaction translators associated with the USB2.0 host controller.  These bits would be set to '0001b' for Multi-Port Host, and '0000b' for Single-Port Host.
23–20 N_PTT	Number of Ports per Transaction Translator (N_PTT). Default value '0000b'  This field indicates the number of ports assigned to each transaction translator within the USB2.0 host controller.  These bits would be set to equal N_PORTS for Multi-Port Host, and '0000b' for Single-Port Host.
19–17 -	This field is reserved. Reserved
16 PI	Port Indicators (P INDICATOR)  This bit indicates whether the ports support port indicator control. When set to one, the port status and control registers include a read/writeable field for controlling the state of the port indicator  This bit is "1b" in all controller core.
15–12 N_CC	Number of Companion Controller (N_CC).  This field indicates the number of companion controllers associated with this USB2.0 host controller.  These bits are '0000b' in all controller core.  0 There is no internal Companion Controller and port-ownership hand-off is not supported. 1 There are internal companion controller(s) and port-ownership hand-offs is supported.
11–8 N_PCC	Number of Ports per Companion Controller  This field indicates the number of ports supported per internal Companion Controller. It is used to indicate the port routing configuration to the system software.  For example, if N_PORTS has a value of 6 and N_CC has a value of 2 then N_PCC could have a value of 3. The convention is that the first N_PCC ports are assumed to be routed to companion controller 1, the next N_PCC ports to companion controller 2, etc. In the previous example, the N_PCC could have been 4, where the first 4 are routed to companion controller 1 and the last two are routed to companion controller 2. The number in this field must be consistent with N_PORTS and N_CC.  These bits are '0000b' in all controller core.
7–5 -	This field is reserved. Reserved
4 PPC	Port Power Control  This field indicates whether the host controller implementation includes port power control. A one indicates the ports have port power switches. A zero indicates the ports do not have port power switches. The value of this field affects the functionality of the Port Power field in each port status and control register
3–0 N_PORTS	Number of downstream ports. This field specifies the number of physical downstream ports implemented on this host controller. The value of this field determines how many port registers are addressable in the Operational Register.  Valid values are in the range of 1h to Fh. A zero in this field is undefined.  These bits are always set to '0001b' because all controller cores are Single-Port Host.

### 52.6.15 Host Controller Capability Parameters (USBC\_n\_HCCPARAMS)

This register identifies multiple mode control (time-base bit functionality), addressing capability.

Address: 218\_4000h base + 108h offset + (512d × i), where i=0d to 2d





USBC\_n\_HCCPARAMS field descriptions

Field	Description
31–16 -	This field is reserved. Reserved
15–8 EECP	EHCI Extended Capabilities Pointer.  This field indicates the existence of a capabilities list. A value of 00h indicates no extended capabilities are implemented. A non-zero value in this register indicates the offset in PCI configuration space of the first EHCI extended capability. The pointer value must be 40h or greater if implemented to maintain the consistency of the PCI header defined for this class of device.  <b>NOTE:</b> These bits are set '00h' in all controller core.
7–4 IST	Isochronous Scheduling Threshold.  This field indicates, relative to the current position of the executing host controller, where software can reliably update the isochronous schedule. When bit [7] is zero, the value of the least significant 3 bits indicates the number of micro-frames a host controller can hold a set of isochronous data structures (one or more) before flushing the state. When bit [7] is a one, then host software assumes the host controller may cache an isochronous data structure for an entire frame.  These bits are set '00h' in all controller core.
3 -	This field is reserved. Reserved

Table continues on the next page...

**USBC\_n\_HCCPARAMS field descriptions (continued)**

Field	Description
2 ASP	<p>Asynchronous Schedule Park Capability</p> <p>If this bit is set to a one, then the host controller supports the park feature for high-speed queue heads in the Asynchronous Schedule. The feature can be disabled or enabled and set to a specific level by using the <i>Asynchronous Schedule Park Mode Enable</i> and <i>Asynchronous Schedule Park Mode Count</i> fields in the USBCMD register.</p> <p><b>NOTE:</b> ASP bit reset value: '00b' for OTG controller core, '11b' for Host-only controller core.</p>
1 PFL	<p>Programmable Frame List Flag</p> <p>If this bit is set to zero, then the system software must use a frame list length of 1024 elements with this host controller. The USBCMD register Frame List Size field is a read-only register and must be set to zero.</p> <p>If set to a one, then the system software can specify and use a smaller frame list and configure the host controller via the USBCMD register Frame List Size field. The frame list must always be aligned on a 4K-page boundary. This requirement ensures that the frame list is always physically contiguous.</p> <p>This bit is set '1b' in all controller core.</p>
0 ADC	<p>64-bit Addressing Capability</p> <p>This bit is set '0b' in all controller core, no 64-bit addressing capability is supported.</p>

## 52.6.16 Device Controller Interface Version (USBC\_n\_DCIVERSION)

This register indicates the two-byte BCD encoding of the device controller interface version number.

Address: 218\_4000h base + 120h offset + (512d × i), where i=0d to 1d

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	DCIVERSION															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

**USBC\_n\_DCIVERSION field descriptions**

Field	Description
15–0 DCIVERSION	<p>Device Controller Interface Version Number</p> <p>Default value is '01h', which means rev0.1.</p>

## 52.6.17 Device Controller Capability Parameters (USBC\_n\_DCCPARAMS)

These fields describe the overall device capability of the controller.

**NOTE**

This register is only available in OTG controller core.

Address: 218\_4000h base + 124h offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								HC	DC	Reserved		DEN			
W																
Reset	0	0	0	0	0	0	0	1	1	0	0	0	1	0	0	0

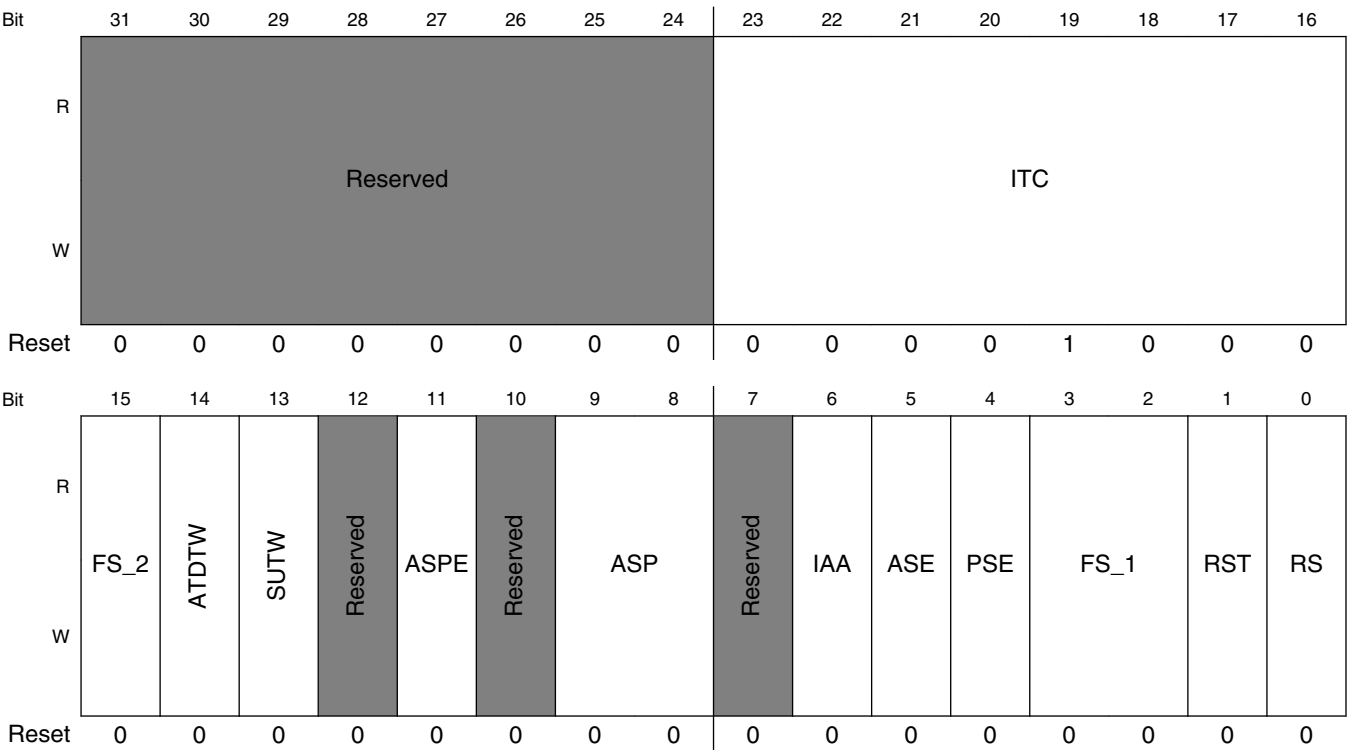
**USBC\_n\_DCCPARAMS field descriptions**

Field	Description
31–9 -	This field is reserved. Reserved
8 HC	Host Capable When this bit is 1, this controller is capable of operating as an EHCI compatible USB 2.0 host controller.
7 DC	Device Capable When this bit is 1, this controller is capable of operating as a USB 2.0 device.
6–5 -	This field is reserved. Reserved
4–0 DEN	Device Endpoint Number This field indicates the number of endpoints built into the device controller. If this controller is not device capable, then this field will be zero. Valid values are 0 - 15.

### 52.6.18 USB Command Register (USBC\_n\_USBCMD)

The Command Register indicates the command to be executed by the serial bus host/device controller. Writing to the register causes a command to be executed.

Address: 218\_4000h base + 140h offset + (512d × i), where i=0d to 2d



USBC\_n\_USBCMD field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 ITC	Interrupt Threshold Control -Read/Write.  The system software uses this field to set the maximum rate at which the host/device controller will issue interrupts. ITC contains the maximum interrupt interval measured in micro-frames. Valid values are shown below.  Value Maximum Interrupt Interval  0x00 Immediate (no threshold) 0x01 1 micro-frame 0x02 2 micro-frames 0x04 4 micro-frames 0x08 8 micro-frames 0x10 16 micro-frames

Table continues on the next page...



## USBC\_n\_USBCMD field descriptions (continued)

Field	Description
	0x20 32 micro-frames 0x40 64 micro-frames
15 FS_2	See also bits 3-2 Frame List Size - (Read/Write or Read Only). [host mode only] This field is Read/Write only if Programmable Frame List Flag in the HCCPARAMS registers is set to one. This field specifies the size of the frame list that controls which bits in the Frame Index Register should be used for the Frame List Current index. <b>NOTE:</b> This field is made up from USBCMD bits 15, 3 and 2. Value Meaning  000 1024 elements (4096 bytes) Default value 001 512 elements (2048 bytes) 010 256 elements (1024 bytes) 011 128 elements (512 bytes) 100 64 elements (256 bytes) 101 32 elements (128 bytes) 110 16 elements (64 bytes) 111 8 elements (32 bytes)
14 ATDTW	Add dTD TripWire - Read/Write. [device mode only] This bit is used as a semaphore to ensure proper addition of a new dTD to an active (primed) endpoint's linked list. This bit is set and cleared by software. This bit would also be cleared by hardware when state machine is hazard region for which adding a dTD to a primed endpoint may go unrecognized.
13 SUTW	Setup TripWire - Read/Write. [device mode only] This bit is used as a semaphore to ensure that the setup data payload of 8 bytes is extracted from a QH by the DCD without being corrupted. If the setup lockout mode is off (SLOM bit in USB core register n_USBMODE, see <a href="#">USB Device Mode (USBC_n_USBMODE)</a> ) then there is a hazard when new setup data arrives while the DCD is copying the setup data payload from the QH for a previous setup packet. This bit is set and cleared by software. This bit would also be cleared by hardware when a hazard detected.
12 -	This field is reserved. Reserved
11 ASPE	Asynchronous Schedule Park Mode Enable - Read/Write. If the <i>Asynchronous Park Capability</i> bit in the HCCPARAMS register is a one, then this bit defaults to a 1h and is R/W. Otherwise the bit must be a zero and is RO. Software uses this bit to enable or disable Park mode. When this bit is one, Park mode is enabled. When this bit is a zero, Park mode is disabled. <b>NOTE:</b> ASPE bit reset value: '0b' for OTG controller core, '1b' for Host-only controller core.
10 -	This field is reserved. Reserved
9-8 ASP	Asynchronous Schedule Park Mode Count - Read/Write. If the <i>Asynchronous Park Capability</i> bit in the HCCPARAMS register is a one, then this field defaults to 3h and is R/W. Otherwise it defaults to zero and is Read-Only. It contains a count of the number of successive transactions the host controller is allowed to execute from a high-speed queue head on the

Table continues on the next page...

## USBC\_n\_USBCMD field descriptions (continued)

Field	Description
	Asynchronous schedule before continuing traversal of the Asynchronous schedule. Valid values are 1h to 3h. Software must not write a zero to this bit when <i>Park Mode Enable</i> is a one as this will result in undefined behavior.  This field is set to 3h in all controller core.
7 -	This field is reserved. Reserved
6 IAA	Interrupt on Async Advance Doorbell - Read/Write.  This bit is used as a doorbell by software to tell the host controller to issue an interrupt the next time it advances asynchronous schedule. Software must write a 1 to this bit to ring the doorbell.  When the host controller has evicted all appropriate cached schedule states, it sets the Interrupt on Async Advance status bit in the USBSTS register. If the Interrupt on Sync Advance Enable bit in the USBINTR register is one, then the host controller will assert an interrupt at the next interrupt threshold.  The host controller sets this bit to zero after it has set the Interrupt on Sync Advance status bit in the USBSTS register to one. Software should not write a one to this bit when the asynchronous schedule is inactive. Doing so will yield undefined results.  This bit is only used in host mode. Writing a one to this bit when device mode is selected will have undefined results.
5 ASE	Asynchronous Schedule Enable - Read/Write. Default 0b.  This bit controls whether the host controller skips processing the Asynchronous Schedule.  Only the host controller uses this bit.  Values Meaning  0 Do not process the Asynchronous Schedule. 1 Use the ASYNCLISTADDR register to access the Asynchronous Schedule.
4 PSE	Periodic Schedule Enable- Read/Write. Default 0b.  This bit controls whether the host controller skips processing the Periodic Schedule.  Only the host controller uses this bit.  Values Meaning  0 Do not process the Periodic Schedule 1 Use the PERIODICLISTBASE register to access the Periodic Schedule.
3-2 FS_1	See description at bit 15
1 RST	Controller Reset (RESET) - Read/Write. Software uses this bit to reset the controller. This bit is set to zero by the Host/Device Controller when the reset process is complete. Software cannot terminate the reset process early by writing a zero to this register.  Host operation mode:  When software writes a one to this bit, the Controller resets its internal pipelines, timers, counters, state machines etc. to their initial value. Any transaction currently in progress on USB is immediately terminated. A USB reset is not driven on downstream ports. Software should not set this bit to a one when the HCHalted bit in the USBSTS register is a zero. Attempting to reset an actively running host controller will result in undefined behavior.  Device operation mode:

*Table continues on the next page...*

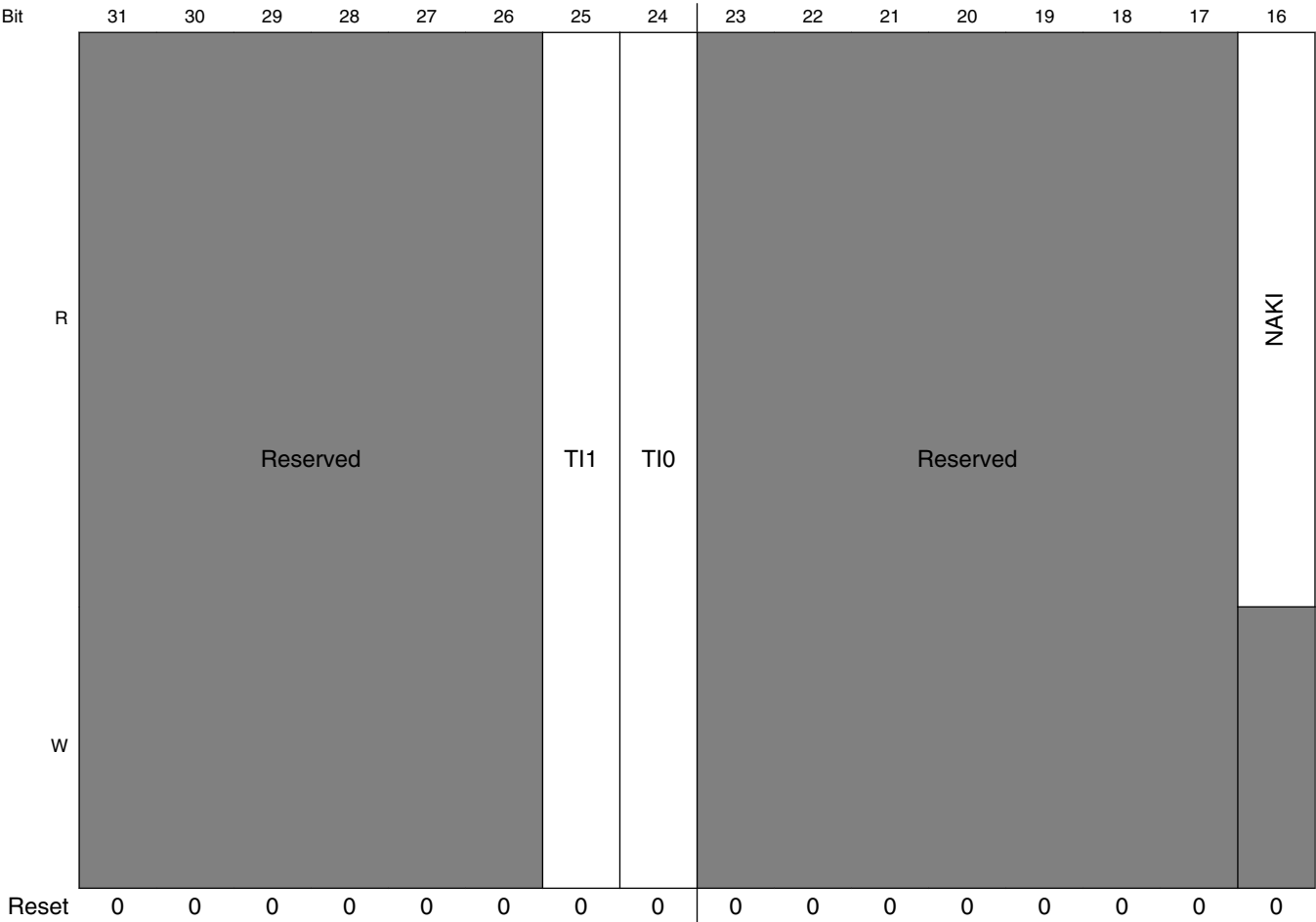
**USBC\_n\_USBCMD field descriptions (continued)**

Field	Description
	When software writes a one to this bit, the Controller resets its internal pipelines, timers, counters, state machines etc. to their initial value. Writing a one to this bit when the device is in the attached state is not recommended, because the effect on an attached host is undefined. In order to ensure that the device is not in an attached state before initiating a device controller reset, all primed endpoints should be flushed and the USBCMD Run/Stop bit should be set to 0.
0 RS	<p>Run/Stop (RS) - Read/Write. Default 0b. 1=Run. 0=Stop.</p> <p>Host operation mode:</p> <p>When set to '1b', the Controller proceeds with the execution of the schedule. The Controller continues execution as long as this bit is set to a one. When this bit is set to 0, the Host Controller completes the current transaction on the USB and then halts. The HC Halted bit in the status register indicates when the Controller has finished the transaction and has entered the stopped state. Software should not write a one to this field unless the controller is in the Halted state (that is, HCHalted in the USBSTS register is a one).</p> <p>Device operation mode:</p> <p>Writing a one to this bit will cause the controller to enable a pull-up on D+ and initiate an attach event. This control bit is not directly connected to the pull-up enable, as the pull-up will become disabled upon transitioning into high-speed mode. Software should use this bit to prevent an attach event before the controller has been properly initialized. Writing a 0 to this will cause a detach event.</p>

### 52.6.19 USB Status Register (USBC\_n\_USBSTS)

This register indicates various states of the Host/Device Controller and any pending interrupts. This register does not indicate status resulting from a transaction on the serial bus.

Address: 218\_4000h base + 144h offset + (512d × i), where i=0d to 2d



Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R					Reserved		Reserved									
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	AS	PS	RCL	HCH		ULPII		SLI	SRI	URI	AAI	SEI	FRI	PCI	UEI	UI

USBC\_n\_USBSTS field descriptions

Field	Description
31–26 -	This field is reserved. Reserved
25 TI1	General Purpose Timer Interrupt 1(GPTINT1)--R/WC. This bit is set when the counter in the GPTIMER1CTRL register transitions to zero, writing a one to this bit will clear it.
24 TI0	General Purpose Timer Interrupt 0(GPTINT0)--R/WC. This bit is set when the counter in the GPTIMER0CTRL register transitions to zero, writing a one to this bit clears it.
23–17 -	This field is reserved. Reserved
16 NAKI	NAK Interrupt Bit--RO. This bit is set by hardware when for a particular endpoint both the TX/RX Endpoint NAK bit and corresponding TX/RX Endpoint NAK Enable bit are set. This bit is automatically cleared by hardware when all Enabled TX/RX Endpoint NAK bits are cleared.
15 AS	Asynchronous Schedule Status - Read Only. This bit reports the current real status of the Asynchronous Schedule. When set to zero the asynchronous schedule status is disabled and if set to one the status is enabled. The Host Controller is not required to immediately disable or enable the Asynchronous Schedule when software transitions the Asynchronous Schedule Enable bit in the USBCMD register. When this bit and the Asynchronous Schedule Enable bit are the same value, the Asynchronous Schedule is either enabled (1) or disabled (0). Only used in the host operation mode.
14 PS	Periodic Schedule Status - Read Only.

Table continues on the next page...

## USBC\_n\_USBSTS field descriptions (continued)

Field	Description
	<p>This bit reports the current real status of the Periodic Schedule. When set to zero the periodic schedule is disabled, and if set to one the status is enabled. The Host Controller is not required to immediately disable or enable the Periodic Schedule when software transitions the Periodic Schedule Enable bit in the USBCMD register. When this bit and the Periodic Schedule Enable bit are the same value, the Periodic Schedule is either enabled (1) or disabled (0).</p> <p>Only used in the host operation mode.</p>
13 RCL	<p>Reclamation - Read Only.</p> <p>This is a read-only status bit used to detect an empty asynchronous schedule.</p> <p>Only used in the host operation mode.</p>
12 HCH	<p>HCHalted - Read Only.</p> <p>This bit is a zero whenever the Run/Stop bit is a one. The Controller sets this bit to one after it has stopped executing because of the Run/Stop bit being set to 0, either by software or by the Controller hardware (for example, an internal error).</p> <p>Only used in the host operation mode.</p> <p>Default value is '0b' for OTG core, and '1b' for Host1/Host2/Host3 core.</p> <p>This is because OTG core is not operating as host in default. Please see CM bit in USB_n_USBMODE register.</p> <p><b>NOTE:</b> HCH bit reset value: '0b' for OTG controller core, '1b' for Host-only controller core.</p>
11 -	<p>This field is reserved.</p> <p>Reserved</p>
10 ULPII	<p>ULPI Interrupt - R/WC.</p> <p>This bit will be set '1b' by hardware when there is an event completion in ULPI viewport.</p> <p>This bit is usable only if the controller support UPLI interface mode.</p>
9 -	<p>This field is reserved.</p> <p>Reserved</p>
8 SLI	<p>DCSuspend - R/WC.</p> <p>When a controller enters a suspend state from an active state, this bit will be set to a one. The device controller clears the bit upon exiting from a suspend state.</p> <p>Only used in device operation mode.</p>
7 SRI	<p>SOF Received - R/WC.</p> <p>When the device controller detects a Start Of (micro) Frame, this bit will be set to a one. When a SOF is extremely late, the device controller will automatically set this bit to indicate that an SOF was expected. Therefore, this bit will be set roughly every 1ms in device FS mode and every 125ms in HS mode and will be synchronized to the actual SOF that is received.</p> <p>Because the device controller is initialized to FS before connect, this bit will be set at an interval of 1ms during the prelude to connect and chirp.</p> <p>In host mode, this bit will be set every 125us and can be used by host controller driver as a time base.</p> <p>Software writes a 1 to this bit to clear it.</p>
6 URI	<p>USB Reset Received - R/WC.</p> <p>When the device controller detects a USB Reset and enters the default state, this bit will be set to a one. Software can write a 1 to this bit to clear the USB Reset Received status bit.</p> <p>Only used in device operation mode.</p>

Table continues on the next page...

**USBC\_n\_USBSTS field descriptions (continued)**

<b>Field</b>	<b>Description</b>
5 AAI	<p>Interrupt on Async Advance - R/WC.</p> <p>System software can force the host controller to issue an interrupt the next time the host controller advances the asynchronous schedule by writing a one to the Interrupt on Async Advance Doorbell bit in the n_USBCMD register. This status bit indicates the assertion of that interrupt source.</p> <p>Only used in host operation mode.</p>
4 SEI	<p>System Error- R/WC.</p> <p>This bit is will be set to '1b' when an Error response is seen to a read on the system interface.</p>
3 FRI	<p>Frame List Rollover - R/WC.</p> <p>The Host Controller sets this bit to a one when the Frame List Index rolls over from its maximum value to zero. The exact value at which the rollover occurs depends on the frame list size. For example. If the frame list size (as programmed in the Frame List Size field of the USB_n_USBCMD register) is 1024, the Frame Index Register rolls over every time FRINDEX [13] toggles. Similarly, if the size is 512, the Host Controller sets this bit to a one every time FHINDEX [12] toggles.</p> <p>Only used in host operation mode.</p>
2 PCI	<p>Port Change Detect - R/WC.</p> <p>The Host Controller sets this bit to a one when on any port a Connect Status occurs, a Port Enable/Disable Change occurs, or the Force Port Resume bit is set as the result of a J-K transition on the suspended port.</p> <p>The Device Controller sets this bit to a one when the port controller enters the full or high-speed operational state. When the port controller exits the full or high-speed operation states due to Reset or Suspend events, the notification mechanisms are the USB Reset Received bit and the DCSuspend bits respectively.</p>
1 UEI	<p>USB Error Interrupt (USBERRINT) - R/WC.</p> <p>When completion of a USB transaction results in an error condition, this bit is set by the Host/Device Controller. This bit is set along with the USBINT bit, if the TD on which the error interrupt occurred also had its interrupt on complete (IOC) bit set</p> <p>The device controller detects resume signaling only.</p>
0 UI	<p>USB Interrupt (USBINT) - R/WC.</p> <p>This bit is set by the Host/Device Controller when the cause of an interrupt is a completion of a USB transaction where the Transfer Descriptor (TD) has an interrupt on complete (IOC) bit set.</p> <p>This bit is also set by the Host/Device Controller when a short packet is detected. A short packet is when the actual number of bytes received was less than the expected number of bytes.</p>

## 52.6.20 Interrupt Enable Register (USBC\_n\_USBINTR)

The interrupts to software are enabled with this register. An interrupt is generated when a bit is set and the corresponding interrupt source is active. The USB Status register (n\_USBSTS) still shows interrupt sources even if they are disabled by the n\_USBINTR register, allowing polling of interrupt events by the software.

Address: 218\_4000h base + 148h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved						TIE1	TIE0	Reserved				UPIE	UAIE	Reserved	NAKE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	-						ULPIE	Reserved	SLE	SRE	URE	AAE	SEE	FRE	PCE	UEE	UE
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**USBC\_n\_USBINTR field descriptions**

Field	Description
31–26 -	This field is reserved. Reserved
25 TIE1	General Purpose Timer #1 Interrupt Enable When this bit is one and the TI1 bit in n_USBSTS register is a one the controller will issue an interrupt.
24 TIE0	General Purpose Timer #0 Interrupt Enable When this bit is one and the TI0 bit in n_USBSTS register is a one the controller will issue an interrupt.
23–20 -	This field is reserved. Reserved
19 UPIE	USB Host Periodic Interrupt Enable

*Table continues on the next page...*



**USBC\_n\_USBINTR field descriptions (continued)**

Field	Description
	When this bit is one, and the UPI bit in the n_USBSTS register is one, host controller will issue an interrupt at the next interrupt threshold.
18 UAIE	USB Host Asynchronous Interrupt Enable When this bit is one, and the UAI bit in the n_USBSTS register is one, host controller will issue an interrupt at the next interrupt threshold.
17 -	This field is reserved. Reserved
16 NAKE	NAK Interrupt Enable When this bit is one and the NAKI bit in n_USBSTS register is a one the controller will issue an interrupt.
15–11 -	These bits are reserved and should be set to zero.
10 ULPIE	ULPI Interrupt Enable When this bit is one and the UPLI bit in n_USBSTS register is a one the controller will issue an interrupt. This bit is usable only if the controller support UPLI interface mode.
9 -	This field is reserved. Reserved
8 SLE	Sleep Interrupt Enable When this bit is one and the SLI bit in n_n_USBSTS register is a one the controller will issue an interrupt. Only used in device operation mode.
7 SRE	SOF Received Interrupt Enable When this bit is one and the SRI bit in n_USBSTS register is a one the controller will issue an interrupt.
6 URE	USB Reset Interrupt Enable When this bit is one and the URI bit in n_USBSTS register is a one the controller will issue an interrupt. Only used in device operation mode.
5 AAE	Async Advance Interrupt Enable When this bit is one and the AAI bit in n_USBSTS register is a one the controller will issue an interrupt. Only used in host operation mode.
4 SEE	System Error Interrupt Enable When this bit is one and the SEI bit in n_USBSTS register is a one the controller will issue an interrupt. Only used in host operation mode.
3 FRE	Frame List Rollover Interrupt Enable When this bit is one and the FRI bit in n_USBSTS register is a one the controller will issue an interrupt. Only used in host operation mode.
2 PCE	Port Change Detect Interrupt Enable When this bit is one and the PCI bit in n_USBSTS register is a one the controller will issue an interrupt.
1 UEE	USB Error Interrupt Enable When this bit is one and the UEI bit in n_USBSTS register is a one the controller will issue an interrupt.
0 UE	USB Interrupt Enable When this bit is one and the UI bit in n_USBSTS register is a one the controller will issue an interrupt.

## 52.6.21 USB Frame Index (USBC\_n\_FRINDEX)

This register is used by the host controller to index the periodic frame list. The register updates every 125 microseconds (once each micro-frame). Bits [N: 3] are used to select a particular entry in the Periodic Frame List during periodic schedule execution. The number of bits used for the index depends on the size of the frame list as set by system software in the Frame List Size field in the n\_USBCMD register.

This register must be written as a DWord. Byte writes produce undefined results. This register cannot be written unless the Host Controller is in the 'Halted' state as indicated by the HCHalted bit. A write to this register while the Run/Stop hit is set to a one produces undefined results. Writes to this register also affect the SOF value.

In device mode this register is read only and, the device controller updates the FRINDEX [13:3] register from the frame number indicated by the SOF marker. Whenever a SOF is received by the USB bus, FRINDEX [13:3] will be checked against the SOF marker. If FRINDEX [13:3] is different from the SOF marker, FRINDEX [13:3] will be set to the SOF value and FRINDEX [2:0] will be set to zero (that is, SOF for 1 ms frame). If FRINDEX [13:3] is equal to the SOF value, FRINDEX [2:0] will be increment (that is, SOF for 125 us micro-frame.).

Address: 218\_4000h base + 14Ch offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																FRINDEX															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### USBC\_n\_FRINDEX field descriptions

Field	Description
31–14 -	This field is reserved. Reserved
13–0 FRINDEX	<p>Frame Index.</p> <p>The value, in this register, increments at the end of each time frame (micro-frame). Bits [N: 3] are used for the Frame List current index. This means that each location of the frame list is accessed 8 times (frames or micro-frames) before moving to the next index.</p> <p>The following illustrates values of N based on the value of the Frame List Size field in the USBCMD register, when used in host mode.</p> <p>USBCMD [Frame List Size] Number Elements N</p> <p>In device mode the value is the current frame number of the last frame transmitted. It is not used as an index.</p> <p>In either mode bits 2:0 indicate the current microframe.</p> <p>000 (1024) 12</p> <p>001 (512) 11</p>

Table continues on the next page...

**USBC\_n\_FRINDEX field descriptions (continued)**

Field	Description
010 (256) 10	
011 (128) 9	
100 (64) 8	
101 (32) 7	
110 (16) 6	
111 (8) 5	

### 52.6.22 Frame List Base Address (USBC\_n\_PERIODICLISTBASE)

Host Controller only

This 32-bit register contains the beginning address of the Periodic Frame List in the system memory. HCD loads this register prior to starting the schedule execution by the Host Controller. The memory structure referenced by this physical memory pointer is assumed to be 4-Kbyte aligned. The contents of this register are combined with the Frame Index Register (USB\_n\_FRINDEX) to enable the Host Controller to step through the Periodic Frame List in sequence.

Address: 218\_4000h base + 154h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**USBC\_n\_PERIODICLISTBASE field descriptions**

Field	Description
31–12 BASEADR	Base Address (Low). These bits correspond to memory address signals [31:12], respectively. Only used by the host controller.
11–0 -	This field is reserved. Reserved

### 52.6.23 Device Address (USBC\_n\_DEVICEADDR)

Device Controller only

The upper seven bits of this register represent the device address. After any controller reset or a USB reset, the device address is set to the default address (0). The default address will match all incoming addresses. Software shall reprogram the address after receiving a SET\_ADDRESS descriptor.

Address: 218\_4000h base + 154h offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	USBADR							USBADRA	Reserved							
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### USBC\_n\_DEVICEADDR field descriptions

Field	Description
31–25 USBADR	Device Address. These bits correspond to the USB device address
24 USBADRA	Device Address Advance. Default=0.  When this bit is '0', any writes to USBADR are instantaneous. When this bit is written to a '1' at the same time or before USBADR is written, the write to the USBADR field is staged and held in a hidden register. After an IN occurs on endpoint 0 and is ACKed, USBADR will be loaded from the holding register.  Hardware will automatically clear this bit on the following conditions: 1) IN is ACKed to endpoint 0. (USBADR is updated from staging register). 2) OUT/SETUP occur to endpoint 0. (USBADR is not updated). 3) Device Reset occurs (USBADR is reset to 0).  <b>NOTE:</b> After the status phase of the SET_ADDRESS descriptor, the DCD has 2 ms to program the USBADR field. This mechanism will ensure this specification is met when the DCD can not write of the device address within 2ms from the SET_ADDRESS status phase. If the DCD writes the USBADR with USBADRA=1 after the SET_ADDRESS data phase (before the prime of the status phase), the USBADR will be programmed instantly at the correct time and meet the 2ms USB requirement.
23–0 -	This field is reserved. Reserved

## 52.6.24 Next Asynch. Address (USBC\_n\_ASYNCLISTADDR)

Host Controller only

This 32-bit register contains the address of the next asynchronous queue head to be executed by the host. Bits [4:0] of this register cannot be modified by the system software and will always return a zero when read.

Address: 218\_4000h base + 158h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ASYBASE																Reserved															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### USBC\_n\_ASYNCLISTADDR field descriptions

Field	Description
31–5 ASYBASE	Link Pointer Low (LPL). These bits correspond to memory address signals [31:5], respectively. This field may only reference a Queue Head (QH). Only used by the host controller.
4–0 -	This field is reserved. Reserved

## 52.6.25 Endpoint List Address (USBC\_n\_ENDPTLISTADDR)

Device Controller only

In device mode, this register contains the address of the top of the endpoint list in system memory. Bits [10:0] of this register cannot be modified by the system software and will always return a zero when read.

The memory structure referenced by this physical memory pointer is assumed 64-byte.

Address: 218\_4000h base + 158h offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	EPBASE																Reserved															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### USBC\_n\_ENDPTLISTADDR field descriptions

Field	Description
31–11 EPBASE	Endpoint List Pointer(Low). These bits correspond to memory address signals [31:11], respectively. This field will reference a list of up to 32 Queue Head (QH) (that is, one queue head per endpoint & direction).
10–0 -	This field is reserved. Reserved

## 52.6.26 Programmable Burst Size (USBC\_n\_BURSTSIZE)

This register is used to control the burst size used during data movement on the AHB master interface. This register is ignored if AHBBRST bits in SBUSCFG register is non-zero value.

Address: 218\_4000h base + 160h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																TXPBURST								RXPBURST							
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

### USBC\_n\_BURSTSIZE field descriptions

Field	Description
31–17 -	This field is reserved. Reserved
16–8 TXPBURST	Programmable TX Burst Size. Default value is determined by TXBURST bits in n_HWTXBUF.  This register represents the maximum length of a the burst in 32-bit words while moving data from system memory to the USB bus.
7–0 RXPBURST	Programmable RX Burst Size. Default value is determined by TXBURST bits in n_HWRXBUF.  This register represents the maximum length of a the burst in 32-bit words while moving data from the USB bus to system memory.

## 52.6.27 TX FIFO Fill Tuning (USBC\_n\_TXFILLTUNING)

The fields in this register control performance tuning associated with how the host controller posts data to the TX latency FIFO before moving the data onto the USB bus. The specific areas of performance include the how much data to post into the FIFO and an estimate for how long that operation should take in the target system.

Definitions:

$T_0$  = Standard packet overhead

$T_1$  = Time to send data payload

$T_{ff}$  = Time to fetch packet into TX FIFO up to specified level.

$T_s$  = Total Packet Flight Time (send-only) packet

$T_s = T_0 + T_1$

$T_p$  = Total Packet Time (fetch and send) packet

$$T_p = T_{ff} + T_0 + T_1$$

Upon discovery of a transmit (OUT/SETUP) packet in the data structures, host controller checks to ensure  $T_p$  remains before the end of the [micro]frame. If so it proceeds to pre-fill the TX FIFO. If at anytime during the pre-fill operation the time remaining the [micro]frame is  $< T_s$  then the packet attempt ceases and the packet is tried at a later time. Although this is not an error condition and the host controller will eventually recover, a mark will be made the scheduler health counter to note the occurrence of a "back-off" event. When a back-off event is detected, the partial packet fetched may need to be discarded from the latency buffer to make room for periodic traffic that will begin after the next SOF. Too many back-off events can waste bandwidth and power on the system bus and thus should be minimized (not necessarily eliminated). Back-offs can be minimized with use of the  $n\_TSCHEALTH$  ( $T_{ff}$ ) described below.

### NOTE

The reset value could vary from instance to instance. Please see the detail in bit field description and ignore reset value in summary table in this case!

Address: 218\_4000h base + 164h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0				
R	Reserved																TXFIFOTHRES						Reserved			TXSCHHEALTH					TXSCHOH					
W																																				
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0				

### USBC\_n\_TXFILLTUNING field descriptions

Field	Description
31–22 -	This field is reserved. Reserved
21–16 TXFIFOTHRES	FIFO Burst Threshold. (Read/Write)  This register controls the number of data bursts that are posted to the TX latency FIFO in host mode before the packet begins on to the bus. The minimum value is 2 and this value should be as low as possible to maximize USB performance. A higher value can be used in systems with unpredictable latency and/or insufficient bandwidth where the FIFO may underrun because the data transferred from the latency FIFO to USB occurs before it can be replenished from system memory. This value is ignored if the Stream Disable bit in USB_n_USBMODE register is set.  Default value is '00h' for OTG controller core, and '02h' for Host-only controller core.
15–13 -	This field is reserved. Reserved
12–8 TXSCHHEALTH	Scheduler Health Counter. (Read/Write To Clear)  This register increments when the host controller fails to fill the TX latency FIFO to the level programmed by TXFIFOTHRES before running out of time to send the packet before the next Start-Of-Frame. This health counter measures the number of times this occurs to provide feedback to selecting a proper TXSCHOH. Writing to this register will clear the counter and this counter will max. at 31.  Default value is '08h' for OTG controller core, and '00h' for Host-only controller core.

Table continues on the next page...

**USBC\_n\_TXFILLTUNING field descriptions (continued)**

Field	Description
7–0 TXSCHOH	<p>Scheduler Overhead. (Read/Write) [Default = 0]</p> <p>This register adds an additional fixed offset to the schedule time estimator described above as Tff. As an approximation, the value chosen for this register should limit the number of back-off events captured in the TXSCHHEALTH to less than 10 per second in a highly utilized bus. Choosing a value that is too high for this register is not desired as it can needlessly reduce USB utilization. The time unit represented in this register is 1.267us when a device is connected in High-Speed Mode. The time unit represented in this register is 6.333us when a device is connected in Low/Full Speed Mode.</p> <p>Default value is '08h' for OTG controller core, and '00h' for Host-only controller core.</p>

**52.6.28 Endpoint NAK (USBC\_n\_ENDPTNAK)**

Address: 218\_4000h base + 178h offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								EPTN								Reserved								EPRN							
W	Reserved								EPTN								Reserved								EPRN							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**USBC\_n\_ENDPTNAK field descriptions**

Field	Description
31–24 -	This field is reserved. Reserved
23–16 EPTN	<p>TX Endpoint NAK - R/WC.</p> <p>Each TX endpoint has 1 bit in this field. The bit is set when the device sends a NAK handshake on a received IN token for the corresponding endpoint.</p> <p>Bit [N] - Endpoint #[N], N is 0-7</p>
15–8 -	This field is reserved. Reserved
7–0 EPRN	<p>RX Endpoint NAK - R/WC.</p> <p>Each RX endpoint has 1 bit in this field. The bit is set when the device sends a NAK handshake on a received OUT or PING token for the corresponding endpoint.</p> <p>Bit [N] - Endpoint #[N], N is 0-7</p>

**52.6.29 Endpoint NAK Enable (USBC\_n\_ENDPTNAKEN)**

Address: 218\_4000h base + 17Ch offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								EPTNE								Reserved								EPRNE							
W	Reserved								EPTNE								Reserved								EPRNE							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	



**USBC\_n\_ENDPTNAKEN field descriptions**

Field	Description
31–24 -	This field is reserved. Reserved
23–16 EPTNE	TX Endpoint NAK Enable - R/W. Each bit is an enable bit for the corresponding TX Endpoint NAK bit. If this bit is set and the corresponding TX Endpoint NAK bit is set, the NAK Interrupt bit is set. Bit [N] - Endpoint #[N], N is 0-7
15–8 -	This field is reserved. Reserved
7–0 EPRNE	RX Endpoint NAK Enable - R/W. Each bit is an enable bit for the corresponding RX Endpoint NAK bit. If this bit is set and the corresponding RX Endpoint NAK bit is set, the NAK Interrupt bit is set. Bit [N] - Endpoint #[N], N is 0-7

**52.6.30 Configure Flag Register (USBC\_n\_CONFIGFLAG)**

Address: 218\_4000h base + 180h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															CF
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

**USBC\_n\_CONFIGFLAG field descriptions**

Field	Description
31–1 -	This field is reserved. Reserved
0 CF	Configure Flag Host software sets this bit as the last action in its process of configuring the Host Controller. This bit controls the default port-routing control logic.  0 Port routing control logic default-routes each port to an implementation dependent classic host controller. 1 Port routing control logic default-routes all ports to this host controller.

### 52.6.31 Port Status & Control (USBC\_n\_PORTSC1)

#### Host Controller

A host controller could implement one to eight port status and control registers. The number is determined by N\_PORTS bits in HWSPARAMs register (please see [Host Controller Structural Parameters \(USBC\\_n\\_HCSPARAMS\)](#) ). Software could read this parameter register to determine how many ports need service.

All controller cores are Single-Port Host, so there is only one port status and control register for each controller core.

This register is only reset by power on reset or controller core reset. The initial conditions of a port are:

- No device connected
- Port disabled

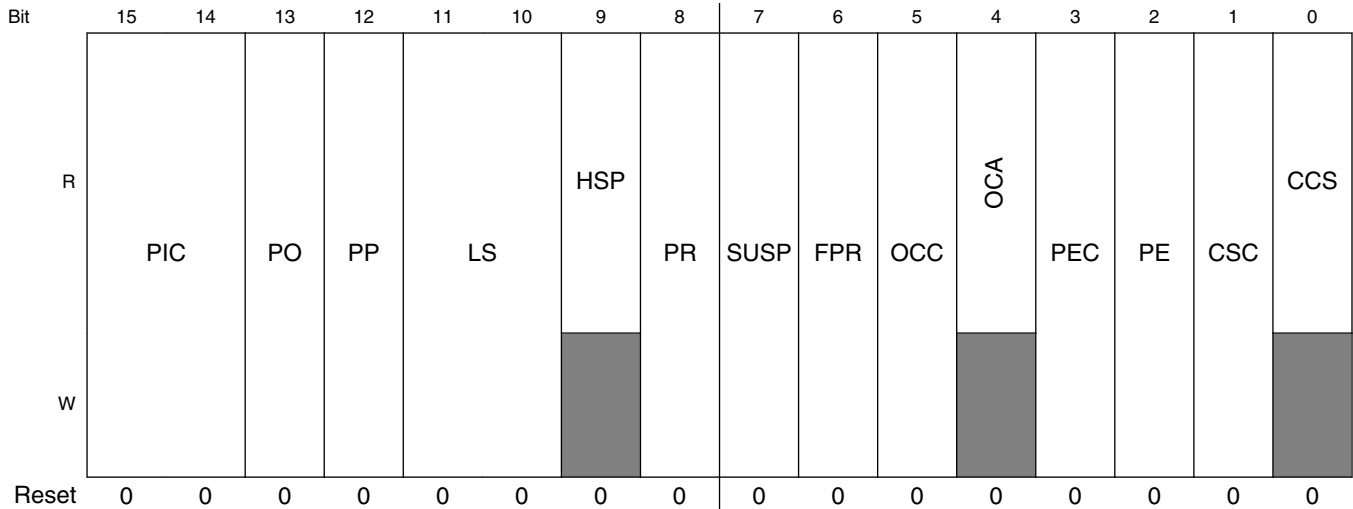
If the port supports power control, this state remains until port power is supplied (by software).

#### Device Controller

A device controller has only port register one (PORTSC1) and it does not support power control. Port control in device mode is only used for status port reset, suspend, and current connect status. It is also used to initiate test mode or force signaling and allows software to put the PHY into low power suspend mode and disable the PHY clock.

Address: 218\_4000h base + 184h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PTS_1		STS	PTW	PSPD		PTS_2	PFSC	PHCD	WKOC	WKDC	WKN	PTC			
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0



USBC\_n\_PORTSC1 field descriptions

Field	Description
31–30 PTS_1	Bit field {bit25, bit31, bit30}: "000b" UTMI/UTMI+ "001b" Reserved "010b" ULPI "011b" Serial/USB 1.1 PHY/IC-USB (FS Only) "100b" HSIC <b>NOTE:</b> All USB port interface modes are listed in this field description, but not all are supported. For detail feature of each controller core, please see <a href="#">Features</a> . The behaviour is unknown when unsupported interface mode is selected.
29 STS	Serial Transceiver Select - Read Only Serial Transceiver Select 1 Serial Interface Engine is selected 0 Parallel Interface signals is selected Serial Interface Engine can be used in combination with UTMI+/ULPI physical interface to provide FS/LS signaling instead of the parallel interface signals. When this bit is set '1b', serial interface engine will be used instead of parallel interface signals. This bit has no effect unless PTS bits is set to select UTMI+/ULPI interface. The Serial/USB1.1 PHY/IC-USB will use the serial interface engine for FS/LS signaling regardless of this bit value.
28 PTW	Parallel Transceiver Width This bit has no effect if serial interface engine is used. For OTG1/OTG2/Host1 core, it is Read-Only. Reset value is '1b'. 0 Select the 8-bit UTMI interface [60MHz] 1 Select the 16-bit UTMI interface [30MHz]
27–26 PSPD	Port Speed - Read Only. This register field indicates the speed at which the port is operating.

Table continues on the next page...

## USBC\_n\_PORTSC1 field descriptions (continued)

Field	Description
	00 Full Speed 01 Low Speed 10 High Speed 11 Undefined
25 PTS_2	See description at bits 31-30
24 PFSC	Port Force Full Speed Connect - Read/Write. Default = 0b.  When this bit is set to '1b', the port will be forced to only connect at Full Speed, It disables the chirp sequence that allows the port to identify itself as High Speed.  1 Forced to full speed 0 Normal operation
23 PHCD	PHY Low Power Suspend - Clock Disable (PLPSCD) - Read/Write. Default = 0b.  When this bit is set to '1b', the PHY clock is disabled. Reading this bit will indicate the status of the PHY clock.  <b>NOTE:</b> The PHY clock cannot be disabled if it is being used as the system clock.  In device mode, The PHY can be put into Low Power Suspend when the device is not running (USBCMD Run/Stop=0b) or the host has signaled suspend (PORTSC1 SUSPEND=1b). PHY Low power suspend will be cleared automatically when the host initials resume. Before forcing a resume from the device, the device controller driver must clear this bit.  In host mode, the PHY can be put into Low Power Suspend when the downstream device has been put into suspend mode or when no downstream device is connected. Low power suspend is completely under the control of software.  1 Disable PHY clock 0 Enable PHY clock
22 WKOC	Wake on Over-current Enable (WKOC_E) - Read/Write. Default = 0b.  Writing this bit to a one enables the port to be sensitive to over-current conditions as wake-up events.  This field is zero if <i>Port Power</i> ( <a href="#">Port Status &amp; Control (USBC_n_PORTSC1)</a> ) is zero.
21 WKDC	Wake on Disconnect Enable (WKDCNNT_E) - Read/Write. Default=0b. Writing this bit to a one enables the port to be sensitive to device disconnects as wake-up events.  This field is zero if <i>Port Power</i> ( <a href="#">Port Status &amp; Control (USBC_n_PORTSC1)</a> ) is zero or in device mode.
20 WKN	Wake on Connect Enable (WKNNT_E) - Read/Write. Default=0b.  Writing this bit to a one enables the port to be sensitive to device connects as wake-up events.  This field is zero if <i>Port Power</i> ( <a href="#">Port Status &amp; Control (USBC_n_PORTSC1)</a> ) is zero or in device mode.
19–16 PTC	Port Test Control - Read/Write. Default = 0000b.  Refer to <a href="#">Port Test Mode</a> for the operational model for using these test modes and the USB Specification Revision 2.0, Chapter 7 for details on each test mode.  The FORCE_ENABLE_FS and FORCE_ENABLE_LS are extensions to the test mode support specified in the EHCI specification. Writing the PTC field to any of the FORCE_ENABLE_{HS/FS/LS} values will force the port into the connected and enabled state at the selected speed. Writing the PTC field back to TEST_MODE_DISABLE will allow the port state machines to progress normally from that point.  <b>NOTE:</b> Low speed operations are not supported as a peripheral device.  Any other value than zero indicates that the port is operating in test mode.

Table continues on the next page...

## USBC\_n\_PORTSC1 field descriptions (continued)

Field	Description
	<p>Value Specific Test</p> <p>0000 TEST_MODE_DISABLE</p> <p>0001 J_STATE</p> <p>0010 K_STATE</p> <p>0011 SE0 (host) / NAK (device)</p> <p>0100 Packet</p> <p>0101 FORCE_ENABLE_HS</p> <p>0110 FORCE_ENABLE_FS</p> <p>0111 FORCE_ENABLE_LS</p> <p>1000-1111 Reserved</p>
15–14 PIC	<p>Port Indicator Control - Read/Write. Default = Ob.</p> <p>Writing to this field has no effect if the P_INDICATOR bit in the HCSPARAMS register is a zero.</p> <p>Refer to the USB Specification Revision 2.0 for a description on how these bits are to be used.</p> <p>This field is zero if <i>Port Power</i> is zero.</p> <p>Bit Value Meaning</p> <p>00 Port indicators are off</p> <p>01 Amber</p> <p>10 Green</p> <p>11 Undefined</p>
13 PO	<p>Port Owner-Read/Write. Default = 0.</p> <p>This bit unconditionally goes to a 0 when the configured bit in the CONFIGFLAG register makes a 0 to 1 transition. This bit unconditionally goes to 1 whenever the Configured bit is zero. System software uses this field to release ownership of the port to a selected host controller (in the event that the attached device is not a high-speed device). Software writes a one to this bit when the attached device is not a high-speed device. A one in this bit means that an internal companion controller owns and controls the port.</p> <p>Port owner handoff is not supported in all controller cores, therefore this bit will always be 0.</p>
12 PP	<p>Port Power (PP)-Read/Write or Read Only.</p> <p>The function of this bit depends on the value of the Port Power Switching (PPC) field in the HCSPARAMS register. The behavior is as follows:</p> <p>PPC PP Operation</p> <p>0 1b <i>Read Only</i> - Host controller does not have port power control switches. Each port is hard-wired to power.</p> <p>1 1b/0b - <i>Read/Write</i>. Host/OTG controller requires port power control switches. This bit represents the current setting of the switch (0=off, 1=on). When power is not available on a port (that is, PP equals a 0), the port is non-functional and will not report attaches, detaches, etc.</p> <p>When an over-current condition is detected on a powered port and PPC is a one, the PP bit in each affected port may be transitional by the host controller driver from a one to a zero (removing power from the port).</p> <p>This feature is implemented in all controller cores (PPC = 1).</p>
11–10 LS	<p>Line Status-Read Only. These bits reflect the current logical levels of the D+ (bit 11) and D- (bit 10) signal lines.</p> <p>In host mode, the use of linestate by the host controller driver is not necessary (unlike EHCI), because the port controller state machine and the port routing manage the connection of LS and FS.</p>

Table continues on the next page...

## USBC\_n\_PORTSC1 field descriptions (continued)

Field	Description
	<p>In device mode, the use of linestate by the device controller driver is not necessary.</p> <p>The encoding of the bits are:</p> <p>Bits [11:10] Meaning</p> <p>00 SE0</p> <p>10 J-state</p> <p>01 K-state</p> <p>11 Undefined</p>
9 HSP	<p>High-Speed Port - Read Only. Default = 0b.</p> <p>When the bit is one, the host/device connected to the port is in high-speed mode and if set to zero, the host/device connected to the port is not in a high-speed mode.</p> <p><b>NOTE:</b> HSP is redundant with PSPD(bit 27, 26) but remained for compatibility.</p>
8 PR	<p>Port Reset - Read/Write or Read Only. Default = 0b.</p> <p>In Host Mode: Read/Write. 1=Port is in Reset. 0=Port is not in Reset. Default 0.</p> <p>When software writes a one to this bit the bus-reset sequence as defined in the USB Specification Revision 2.0 is started. <i>This bit will automatically change to zero after the reset sequence is complete. This behavior is different from EHCI where the host controller driver is required to set this bit to a zero after the reset duration is timed in the driver.</i></p> <p>In Device Mode: This bit is a read only status bit. Device reset from the USB bus is also indicated in the USBSTS register.</p> <p>This field is zero if <i>Port Power</i>(<a href="#">Port Status &amp; Control (USBC_n_PORTSC1)</a>) is zero.</p>
7 SUSP	<p>Suspend - Read/Write or Read Only. Default = 0b.</p> <p>1=Port in suspend state. 0=Port not in suspend state.</p> <p>In Host Mode: Read/Write.</p> <p>Port Enabled Bit and Suspend bit of this register define the port states as follows:</p> <p>Bits [Port Enabled, Suspend] Port State</p> <p>0x Disable</p> <p>10 Enable</p> <p>11 Suspend</p> <p>When in suspend state, downstream propagation of data is blocked on this port, except for port reset. The blocking occurs at the end of the current transaction if a transaction was in progress when this bit was written to 1. In the suspend state, the port is sensitive to resume detection. Note that the bit status does not change until the port is suspended and that there may be a delay in suspending a port if there is a transaction currently in progress on the</p> <p>The host controller will unconditionally set this bit to zero when software sets the <i>Force Port Resume</i> bit to zero. The host controller ignores a write of zero to this bit.</p> <p>If host software sets this bit to a one when the port is not enabled (that is, <i>Port enabled</i> bit is a zero) the results are undefined.</p> <p>This field is zero if <i>Port Power</i>(<a href="#">Port Status &amp; Control (USBC_n_PORTSC1)</a>) is zero in host mode.</p> <p>In Device Mode: Read Only.</p> <p>In device mode this bit is a read only status bit.</p>
6 FPR	<p>Force Port Resume -Read/Write. 1= Resume detected/driven on port. 0=No resume (K-state) detected/driven on port. Default = 0.</p>

Table continues on the next page...

## USBC\_n\_PORTSC1 field descriptions (continued)

Field	Description
	<p>In Host Mode:</p> <p>Software sets this bit to one to drive resume signaling. The Host Controller sets this bit to one if a J-to-K transition is detected while the port is in the Suspend state. When this bit transitions to a one because a J-to-K transition is detected, the <i>Port Change Detect</i> bit in the USBSTS register is also set to one. <i>This bit will automatically change to zero after the resume sequence is complete. This behavior is different from EHCI where the host controller driver is required to set this bit to a zero after the resume duration is timed in the driver.</i></p> <p>Note that when the Host controller owns the port, the resume sequence follows the defined sequence documented in the USB Specification Revision 2.0. The resume signaling (Full-speed 'K') is driven on the port as long as this bit remains a one. This bit will remain a one until the port has switched to the high-speed idle. Writing a zero has no effect because the port controller will time the resume operation clear the bit the port control state switches to HS or FS idle.</p> <p>This field is zero if <i>Port Power</i>(<a href="#">Port Status &amp; Control (USBC_n_PORTSC1)</a>) is zero in host mode.</p> <p>This bit is not-EHCI compatible.</p> <p>In Device mode:</p> <p>After the device has been in Suspend State for 5ms or more, software must set this bit to one to drive resume signaling before clearing. The Device Controller will set this bit to one if a J-to-K transition is detected while the port is in the Suspend state. The bit will be cleared when the device returns to normal operation. Also, when this bit will be cleared because a K-to-J transition detected, the <i>Port Change Detect</i> bit in the USBSTS register is also set to one.</p>
5 OCC	<p>Over-current Change-R/WC. Default=0.</p> <p>This bit is set '1b' by hardware when there is a change to Over-current Active. Software can clear this bit by writing a one to this bit position.</p>
4 OCA	<p>Over-current Active-Read Only. Default 0.</p> <p>This bit will automatically transition from one to zero when the over current condition is removed.</p> <p>1 This port currently has an over-current condition 0 This port does not have an over-current condition.</p>
3 PEC	<p>Port Enable/Disable Change-R/WC. 1=Port enabled/disabled status has changed. 0=No change. Default = 0.</p> <p>In Host Mode:</p> <p>For the root hub, this bit is set to a one only when a port is disabled due to disconnect on the port or due to the appropriate conditions existing at the EOF2 point (See Chapter 11 of the USB Specification). Software clears this by writing a one to it.</p> <p>This field is zero if <i>Port Power</i>(<a href="#">Port Status &amp; Control (USBC_n_PORTSC1)</a>) is zero.</p> <p>In Device mode:</p> <p>The device port is always enabled, so this bit is always '0b'.</p>
2 PE	<p>Port Enabled/Disabled-Read/Write. 1=Enable. 0=Disable. Default 0.</p> <p>In Host Mode:</p> <p>Ports can only be enabled by the host controller as a part of the reset and enable. Software cannot enable a port by writing a one to this field. Ports can be disabled by either a fault condition (disconnect event or other fault condition) or by the host software. Note that the bit status does not change until the port state actually changes. There may be a delay in disabling or enabling a port due to other host controller and bus events.</p> <p>When the port is disabled, (0b) downstream propagation of data is blocked except for reset.</p>

Table continues on the next page...

**USBC\_n\_PORTSC1 field descriptions (continued)**

Field	Description
	<p>This field is zero if <i>Port Power</i>(<a href="#">Port Status &amp; Control (USBC_n_PORTSC1)</a>) is zero in host mode.</p> <p>In Device Mode:</p> <p>The device port is always enabled, so this bit is always '1b'.</p>
1 CSC	<p>Connect Status Change-R/WC. 1 =Change in Current Connect Status. 0=No change. Default 0.</p> <p>In Host Mode:</p> <p>Indicates a change has occurred in the port's Current Connect Status. The host/device controller sets this bit for all changes to the port device connect status, even if system software has not cleared an existing connect status change. For example, the insertion status changes twice before system software has cleared the changed condition, hub hardware will be 'setting' an already-set bit (that is, the bit will remain set). Software clears this bit by writing a one to it.</p> <p>This field is zero if <i>Port Power</i>(<a href="#">Port Status &amp; Control (USBC_n_PORTSC1)</a>) is zero in host mode.</p> <p>In Device Mode:</p> <p>This bit is undefined in device controller mode.</p>
0 CCS	<p>Current Connect Status-Read Only.</p> <p>In Host Mode:</p> <p>1=Device is present on port. 0=No device is present. Default = 0. This value reflects the current state of the port, and may not correspond directly to the event that caused the <i>Connect Status Change</i> bit (Bit 1) to be set.</p> <p>This field is zero if <i>Port Power</i>(<a href="#">Port Status &amp; Control (USBC_n_PORTSC1)</a>) is zero in host mode.</p> <p>In Device Mode:</p> <p>1=Attached. 0=Not Attached. Default=0. A one indicates that the device successfully attached and is operating in either high speed or full speed as indicated by the High Speed Port bit in this register. A zero indicates that the device did not attach successfully or was forcibly disconnected by the software writing a zero to the Run bit in the USBCMD register. It does not state the device being disconnected or suspended.</p>

**52.6.32 On-The-Go Status & control (USBC\_n\_OTGSC)**

This register is available only in OTG controller core. It has four sections:

- OTG Interrupt enables (Read/Write)
- OTG Interrupt status (Read/Write to Clear)
- OTG Status inputs (Read Only)
- OTG Controls (Read/Write)

The status inputs are debounced using a 1 ms time constant. Values on the status inputs that do not persist for more than 1 ms does not cause an update of the status input register, or cause an OTG interrupt.

See also [USB Device Mode \(USBC\\_n\\_USBMODE\)](#) register.



Address: 218\_4000h base + 1A4h offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	DPIE	1msE	BSEIE	BSVIE	ASVIE	AVVIE	IDIE	Reserved	DPIS	1msS	BSEIS	BSVIS	ASVIS	AWVIS	IDIS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	DPS	1msT	BSE	BSV	ASV	AWV	ID	Reserved	IDPU	DP	OT	Reserved	VC	VD	
W																
Reset	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0	0

## USBC\_n\_OTGSC field descriptions

Field	Description
31 -	This field is reserved. Reserved
30 DPIE	Data Pulse Interrupt Enable
29 1msE	1 millisecond timer Interrupt Enable - Read/Write
28 BSEIE	B Session End Interrupt Enable - Read/Write. Setting this bit enables the B session end interrupt.
27 BSVIE	B Session Valid Interrupt Enable - Read/Write. Setting this bit enables the B session valid interrupt.
26 ASVIE	A Session Valid Interrupt Enable - Read/Write. Setting this bit enables the A session valid interrupt.
25 AVVIE	A VBus Valid Interrupt Enable - Read/Write. Setting this bit enables the A VBus valid interrupt.
24 IDIE	USB ID Interrupt Enable - Read/Write. Setting this bit enables the USB ID interrupt.
23 -	This field is reserved. Reserved
22 DPIS	Data Pulse Interrupt Status - Read/Write to Clear. This bit is set when data bus pulsing occurs on DP or DM. Data bus pulsing is only detected when USBMODE.CM = Host (11) and PORTSC1(0)[PP] = 0. Software must write a one to clear this bit.
21 1msS	1 millisecond timer Interrupt Status - Read/Write to Clear. This bit is set once every millisecond. Software must write a one to clear this bit.
20 BSEIS	B Session End Interrupt Status - Read/Write to Clear. This bit is set when VBus has fallen below the B session end threshold. Software must write a one to clear this bit
19 BSVIS	B Session Valid Interrupt Status - Read/Write to Clear. This bit is set when VBus has either risen above or fallen below the B session valid threshold. Software must write a one to clear this bit.
18 ASVIS	A Session Valid Interrupt Status - Read/Write to Clear. This bit is set when VBus has either risen above or fallen below the A session valid threshold. Software must write a one to clear this bit.
17 AVVIS	A VBus Valid Interrupt Status - Read/Write to Clear. This bit is set when VBus has either risen above or fallen below the VBus valid threshold on an A device. Software must write a one to clear this bit.
16 IDIS	USB ID Interrupt Status - Read/Write. This bit is set when a change on the ID input has been detected. Software must write a one to clear this bit.

*Table continues on the next page...*

**USBC\_n\_OTGSC field descriptions (continued)**

<b>Field</b>	<b>Description</b>
15 -	This field is reserved. Reserved
14 DPS	Data Bus Pulsing Status - Read Only. A '1' indicates data bus pulsing is being detected on the port.
13 1msT	1 millisecond timer toggle - Read Only. This bit toggles once per millisecond.
12 BSE	B Session End - Read Only. Indicates VBus is below the B session end threshold.
11 BSV	B Session Valid - Read Only. Indicates VBus is above the B session valid threshold.
10 ASV	A Session Valid - Read Only. Indicates VBus is above the A session valid threshold.
9 AVV	A VBus Valid - Read Only. Indicates VBus is above the A VBus valid threshold.
8 ID	USB ID - Read Only. 0 = A device, 1 = B device
7-6 -	This field is reserved. Reserved
5 IDPU	ID Pullup - Read/Write This bit provide control over the ID pull-up resistor; 0 = off, 1 = on [default]. When this bit is 0, the ID input will not be sampled.
4 DP	Data Pulsing - Read/Write. Setting this bit causes the pullup on DP to be asserted for data pulsing during SRP.
3 OT	OTG Termination - Read/Write. This bit must be set when the OTG device is in device mode, this controls the pulldown on DM.
2 -	This field is reserved. Reserved
1 VC	VBUS Charge - Read/Write. Setting this bit causes the VBus line to be charged. This is used for VBus pulsing during SRP.
0 VD	VBUS_Discharge - Read/Write. Setting this bit causes VBus to discharge through a resistor.

## 52.6.33 USB Device Mode (USBC\_n\_USBMODE)

Address: 218\_4000h base + 1A8h offset + (512d × i), where i=0d to 2d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved	Reserved										SDIS	SLOW	ES	CM	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**USBC\_n\_USBMODE field descriptions**

Field	Description
31–16 -	This field is reserved. Reserved
15 -	This field is reserved. Reserved
14–5 -	This field is reserved. Reserved
4 SDIS	<p>Stream Disable Mode. (0 - Inactive [default]; 1 - Active)</p> <p>Device Mode: Setting to a '1' disables double priming on both RX and TX for low bandwidth systems. This mode ensures that when the RX and TX buffers are sufficient to contain an entire packet that the standard double buffering scheme is disabled to prevent overruns/underruns in bandwidth limited systems. Note: In High Speed Mode, all packets received are responded to with a NYET handshake when stream disable is active.</p> <p>Host Mode: Setting to a '1' ensures that overruns/underruns of the latency FIFO are eliminated for low bandwidth systems where the RX and TX buffers are sufficient to contain the entire packet. Enabling stream disable also has the effect of ensuring the TX latency is filled to capacity before the packet is launched onto the USB.</p> <p><b>NOTE:</b> Time duration to pre-fill the FIFO becomes significant when stream disable is active. See <a href="#">TX FIFO Fill Tuning (USBC_n_TXFILLTUNING)</a> and TXTTFILLTUNING [MPH Only] to characterize the adjustments needed for the scheduler when using this feature.</p>

*Table continues on the next page...*

**USBC\_n\_USBMODE field descriptions (continued)**

Field	Description
	<b>NOTE:</b> The use of this feature substantially limits the overall USB performance that can be achieved.
3 SLOM	Setup Lockout Mode. In device mode, this bit controls behavior of the setup lock mechanism. See <a href="#">Control Endpoint Operation Model</a> .  0 Setup Lockouts On (default); 1 Setup Lockouts Off (DCD requires use of Setup Data Buffer Tripwire in <a href="#">USB Command Register (USBC_n_USBCMD)</a> ).
2 ES	Endian Select - Read/Write. This bit can change the byte alignment of the transfer buffers to match the host microprocessor. The bit fields in the microprocessor interface and the data structures are unaffected by the value of this bit because they are based upon the 32-bit word.  Bit Meaning  0 Little Endian [Default] 1 Big Endian
1–0 CM	Controller Mode - R/WO. Controller mode is defaulted to the proper mode for host only and device only implementations. For those designs that contain both host & device capability, the controller defaults to an idle state and needs to be initialized to the desired operating mode after reset. For combination host/device controllers, this register can only be written once after reset. If it is necessary to switch modes, software must reset the controller by writing to the <i>RESET</i> bit in the USBCMD register before reprogramming this register.  For OTG controller core, reset value is '00b'.  For Host-only controller core, reset value is '11b'.  00 Idle [Default for combination host/device] 01 Reserved 10 Device Controller [Default for device only controller] 11 Host Controller [Default for host only controller]

**52.6.34 Endpoint Setup Status (USBC\_n\_ENDPTSETUPSTAT)**

Address: 218\_4000h base + 1ACh offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved																ENDPTSETUPSTAT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**USBC\_n\_ENDPTSETUPSTAT field descriptions**

Field	Description
31–16 -	This field is reserved. Reserved
15–0 ENDPTSETUPSTAT	Setup Endpoint Status. For every setup transaction that is received, a corresponding bit in this register is set to one. Software must clear or acknowledge the setup transfer by writing a one to a respective bit after it has read the setup data from Queue head. The response to a setup packet as in the order of operations and total response time is crucial to limit bus time outs while the setup lock our mechanism is engaged. See <a href="#">Managing Endpoints</a> in the Device Operational Model.  This register is only used in device mode.

## 52.6.35 Endpoint Prime (USBC\_n\_ENDPTPRIME)

This register is only used in device mode.

When software sets the prime bit for a given endpoint, the device controller loads the transfer descriptor, pointed to by the queue head, such that the endpoint is ready to transmit or receive when the host sends a request (IN/OUT token). The endpoint will NAK all requests from the host until the endpoint is primed. The controller will automatically re-prime the endpoint with a new transfer descriptor when one is found via the next\_dtd pointer of the current transfer descriptor. Hence, the prime bit must only be set by software when a descriptor is added to the queue head.

Address: 218\_4000h base + 1B0h offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								PETB								Reserved								PERB							
W	Reserved								PETB								Reserved								PERB							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### USBC\_n\_ENDPTPRIME field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 PETB	Prime Endpoint Transmit Buffer - R/WS. For each endpoint a corresponding bit is used to request that a buffer is prepared for a transmit operation in order to respond to a USB IN/INTERRUPT transaction. Software should write a one to the corresponding bit when posting a new transfer descriptor to an endpoint queue head. Hardware automatically uses this bit to begin parsing for a new transfer descriptor from the queue head and prepare a transmit buffer. Hardware clears this bit when the associated endpoint(s) is (are) successfully primed.  <b>NOTE:</b> These bits are momentarily set by hardware during hardware re-priming operations when a dTD is retired, and the dQH is updated.  PETB[N] - Endpoint #N, N is in 0..7
15–8 -	This field is reserved. Reserved
7–0 PERB	Prime Endpoint Receive Buffer - R/WS. For each endpoint, a corresponding bit is used to request a buffer prepare for a receive operation for when a USB host initiates a USB OUT transaction. Software should write a one to the corresponding bit whenever posting a new transfer descriptor to an endpoint queue head. Hardware automatically uses this bit to begin parsing for a new transfer descriptor from the queue head and prepare a receive buffer. Hardware clears this bit when the associated endpoint(s) is (are) successfully primed.  <b>NOTE:</b> These bits are momentarily set by hardware during hardware re-priming operations when a dTD is retired, and the dQH is updated.  PERB[N] - Endpoint #N, N is in 0..7

### 52.6.36 Endpoint Flush (USBC\_n\_ENDPTFLUSH)

This register is only used in device mode.

Address: 218\_4000h base + 1B4h offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								FETB								Reserved								FERB							
W	Reserved								FETB								Reserved								FERB							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### USBC\_n\_ENDPTFLUSH field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 FETB	Flush Endpoint Transmit Buffer - R/WS. Writing one to a bit(s) in this register causes the associated endpoint(s) to clear any primed buffers. If a packet is in progress for one of the associated endpoints, then that transfer continues until completion. Hardware clears this register after the endpoint flush operation is successful.  FETB[N] - Endpoint #N, N is in 0..7
15–8 -	This field is reserved. Reserved
7–0 FERB	Flush Endpoint Receive Buffer - R/WS. Writing one to a bit(s) causes the associated endpoint(s) to clear any primed buffers. If a packet is in progress for one of the associated endpoints, then that transfer continues until completion. Hardware clears this register after the endpoint flush operation is successful.  FERB[N] - Endpoint #N, N is in 0..7

### 52.6.37 Endpoint Status (USBC\_n\_ENDPTSTAT)

This register is only used in device mode.

Address: 218\_4000h base + 1B8h offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								ETBR								Reserved								ERBR							
W	Reserved																Reserved															
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## USBC\_n\_ENDPTSTAT field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 ETBR	Endpoint Transmit Buffer Ready -- Read Only. One bit for each endpoint indicates status of the respective endpoint buffer. This bit is set to one by the hardware as a response to receiving a command from a corresponding bit in the ENDPTPRIME register. There is always a delay between setting a bit in the ENDPTPRIME register and endpoint indicating ready. This delay time varies based upon the current USB traffic and the number of bits set in the ENDPTPRIME register. Buffer ready is cleared by USB reset, by the USB DMA system, or through the ENDPTFLUSH register.  <b>NOTE:</b> These bits are momentarily cleared by hardware during hardware endpoint re-priming operations when a dTD is retired, and the dQH is updated.  ETBR[N] - Endpoint #N, N is in 0..7
15–8 -	This field is reserved. Reserved
7–0 ERBR	Endpoint Receive Buffer Ready -- Read Only. One bit for each endpoint indicates status of the respective endpoint buffer. This bit is set to a one by the hardware as a response to receiving a command from a corresponding bit in the ENDPTPRIME register. There is always a delay between setting a bit in the ENDPTPRIME register and endpoint indicating ready. This delay time varies based upon the current USB traffic and the number of bits set in the ENDPTPRIME register. Buffer ready is cleared by USB reset, by the USB DMA system, or through the ENDPTFLUSH register.  <b>NOTE:</b> These bits are momentarily cleared by hardware during hardware endpoint re-priming operations when a dTD is retired, and the dQH is updated.  ERBR[N] - Endpoint #N, N is in 0..7

## 52.6.38 Endpoint Complete (USBC\_n\_ENDPTCOMPLETE)

This register is only used in device mode.

Address: 218\_4000h base + 1BCh offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								ETCE								Reserved								ERCE							
W	Reserved								ETCE								Reserved								ERCE							
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## USBC\_n\_ENDPTCOMPLETE field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23–16 ETCE	Endpoint Transmit Complete Event - R/WC. Each bit indicates a transmit event (IN/INTERRUPT) occurred and software should read the corresponding endpoint queue to determine the endpoint status. If the corresponding IOC bit is set in the Transfer Descriptor, then this bit is set simultaneously with the <i>USBINT</i> . Writing one clears the corresponding bit in this register.  ETCE[N] - Endpoint #N, N is in 0..7

Table continues on the next page...



**USBC\_n\_ENDPTCOMPLETE field descriptions (continued)**

Field	Description
15–8 -	This field is reserved. Reserved
7–0 ERCE	Endpoint Receive Complete Event - RW/C. Each bit indicates a received event (OUT/SETUP) occurred and software should read the corresponding endpoint queue to determine the transfer status. If the corresponding IOC bit is set in the Transfer Descriptor, then this bit is set simultaneously with the <i>USBINT</i> . Writing one clears the corresponding bit in this register.  ERCE[N] - Endpoint #N, N is in 0..7

**52.6.39 Endpoint Control0 (USBC\_n\_ENDPTCTRL0)**

Every Device implements Endpoint 0 as a control endpoint.

Address: 218\_4000h base + 1C0h offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	Reserved								TXE	Reserved				TXT		Reserved	TXS
W																	
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
R	Reserved								RXE	Reserved				RXT		Reserved	RXS
W																	
Reset	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	

**USBC\_n\_ENDPTCTRL0 field descriptions**

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 1 Enabled

*Table continues on the next page...*

**USBC\_n\_ENDPTCTRL0 field descriptions (continued)**

Field	Description
	Endpoint0 is always enabled.
22–20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 - Control Endpoint0 is fixed as a Control End Point.
17 -	This field is reserved. Reserved
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK [Default] 1 End Point Stalled  Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues returning STALL until the bit is cleared by software or it is automatically cleared upon receipt of a new SETUP request.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 1 Enabled Endpoint0 is always enabled.
6–4 -	This field is reserved. Reserved
3–2 RXT	RX Endpoint Type - Read/Write 00 Control Endpoint0 is fixed as a Control End Point.
1 -	This field is reserved. Reserved
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled  Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues returning STALL until the bit is cleared by software or it is automatically cleared upon receipt of a new SETUP request.

**52.6.40 Endpoint Control 1 (USBC\_n\_ENDPTCTRL1)**

This is endpoint control register for endpoint 1 in device operation mode.

**NOTE**

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type

(that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: 218\_4000h base + 1C4h offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### USBC\_n\_ENDPTCTRL1 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved

Table continues on the next page...

## USBC\_n\_ENDPTCTRL1 field descriptions (continued)

Field	Description
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled  This bit is set automatically upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It is cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint.  Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues to returning STALL until this bit is either cleared by software or automatically cleared as above.  <b>NOTE:</b> For CONTROL type endpoint, there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. Take care that the STALL bit is not set immediately after writing a '1' to it. Please follow this procedure: continually write this STALL bit until it is set or until a new setup has been received by checking the associated ENDPTSETUPSTAT bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 0 Disabled [Default] 1 Enabled  An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence  Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled  This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3–2 RXT	RX Endpoint Type - Read/Write 00 Control

*Table continues on the next page...*

**USBC\_n\_ENDPTCTRL1 field descriptions (continued)**

Field	Description
	01 Isochronous 10 Bulk 11 Reserved
1 RXD	RX Endpoint Data Sink - Read/Write - TBD 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled  This bit is set automatically upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It is cleared automatically upon receipt a SETUP request if this Endpoint is configured as a Control Endpoint,  Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues to returning STALL until this bit is either cleared by software or automatically cleared as above.

**52.6.41 Endpoint Control 2 (USBC\_n\_ENDPTCTRL2)**

This is endpoint control register for endpoint 2 in device operation mode.

**NOTE**

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: 218\_4000h base + 1C8h offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## USB Core Memory Map/Register Definition

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### USBC\_n\_ENDPTCTRL2 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled

Table continues on the next page...

## USBC\_n\_ENDPTCTRL2 field descriptions (continued)

Field	Description
	<p>This bit is set automatically upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It is cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint.</p> <p>Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues to returning STALL until this bit is either cleared by software or automatically cleared as above.</p> <p><b>NOTE:</b> For CONTROL type endpoint, there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. Take care that the STALL bit is not set immediately after writing a '1' to it. Please follow this procedure: continually write this STALL bit until it is set or until a new setup has ben received by checking the associated ENDPTSETUPSTAT bit.</p>
15–8 -	This field is reserved. Reserved
7 RXE	<p>RX Endpoint Enable</p> <p>0 Disabled [Default]</p> <p>1 Enabled</p> <p>An Endpoint should be enabled only after it has been configured.</p>
6 RXR	<p>RX Data Toggle Reset (WS)</p> <p>Write 1 - Reset PID Sequence</p> <p>Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.</p>
5 RXI	<p>RX Data Toggle Inhibit</p> <p>0 Disabled [Default]</p> <p>1 Enabled</p> <p>This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.</p>
4 -	This field is reserved. Reserved.
3–2 RXT	<p>RX Endpoint Type - Read/Write</p> <p>00 Control</p> <p>01 Isochronous</p> <p>10 Bulk</p> <p>11 Reserved</p>
1 RXD	<p>RX Endpoint Data Sink - Read/Write - TBD</p> <p>0 Dual Port Memory Buffer/DMA Engine [Default]</p> <p>Should always be written as zero.</p>
0 RXS	<p>RX Endpoint Stall - Read/Write</p> <p>0 End Point OK. [Default]</p> <p>1 End Point Stalled</p> <p>This bit is set automatically upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It is cleared automatically upon receipt a SETUP request if this Endpoint is configured as a Control Endpoint,</p>

*Table continues on the next page...*

**USBC\_n\_ENDPTCTRL2 field descriptions (continued)**

Field	Description
	Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues to returning STALL until this bit is either cleared by software or automatically cleared as above.

**52.6.42 Endpoint Control 3 (USBC\_n\_ENDPTCTRL3)**

This is endpoint control register for endpoint 3 in device operation mode.

**NOTE**

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: 218\_4000h base + 1CCh offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**USBC\_n\_ENDPTCTRL3 field descriptions**

Field	Description
31–24 -	This field is reserved. Reserved

Table continues on the next page...



**USBC\_n\_ENDPTCTRL3 field descriptions (continued)**

Field	Description
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It is cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues to returning STALL until this bit is either cleared by software or automatically cleared as above. <b>NOTE:</b> For CONTROL type endpoint, there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. Take care that the STALL bit is not set immediately after writing a '1' to it. Please follow this procedure: continually write this STALL bit until it is set or until a new setup has ben received by checking the associated ENDPTSETUPSTAT bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 0 Disabled [Default]

*Table continues on the next page...*

**USBC\_n\_ENDPTCTRL3 field descriptions (continued)**

Field	Description
	1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3-2 RXT	RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Reserved
1 RXD	RX Endpoint Data Sink - Read/Write - TBD 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It is cleared automatically upon receipt a SETUP request if this Endpoint is configured as a Control Endpoint, Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues to returning STALL until this bit is either cleared by software or automatically cleared as above.

**52.6.43 Endpoint Control 4 (USBC\_n\_ENDPTCTRL4)**

This is endpoint control register for endpoint 4 in device operation mode.

**NOTE**

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control

causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: 218\_4000h base + 1D0h offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### USBC\_n\_ENDPTCTRL4 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved

Table continues on the next page...

## USBC\_n\_ENDPTCTRL4 field descriptions (continued)

Field	Description
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled  This bit is set automatically upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It is cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint.  Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues to returning STALL until this bit is either cleared by software or automatically cleared as above.  <b>NOTE:</b> For CONTROL type endpoint, there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. Take care that the STALL bit is not set immediately after writing a '1' to it. Please follow this procedure: continually write this STALL bit until it is set or until a new setup has been received by checking the associated ENDPTSETUPSTAT bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 0 Disabled [Default] 1 Enabled  An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence  Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled  This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3–2 RXT	RX Endpoint Type - Read/Write 00 Control

Table continues on the next page...

**USBC\_n\_ENDPTCTRL4 field descriptions (continued)**

Field	Description
	01 Isochronous 10 Bulk 11 Reserved
1 RXD	RX Endpoint Data Sink - Read/Write - TBD 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled  This bit is set automatically upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It is cleared automatically upon receipt a SETUP request if this Endpoint is configured as a Control Endpoint,  Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues to returning STALL until this bit is either cleared by software or automatically cleared as above.

**52.6.44 Endpoint Control 5 (USBC\_n\_ENDPTCTRL5)**

This is endpoint control register for endpoint 5 in device operation mode.

**NOTE**

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: 218\_4000h base + 1D4h offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## USB Core Memory Map/Register Definition

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### USBC\_n\_ENDPTCTRL5 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled

Table continues on the next page...

## USBC\_n\_ENDPTCTRL5 field descriptions (continued)

Field	Description
	<p>This bit is set automatically upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It is cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint.</p> <p>Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues to returning STALL until this bit is either cleared by software or automatically cleared as above.</p> <p><b>NOTE:</b> For CONTROL type endpoint, there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. Take care that the STALL bit is not set immediately after writing a '1' to it. Please follow this procedure: continually write this STALL bit until it is set or until a new setup has ben received by checking the associated ENDPTSETUPSTAT bit.</p>
15–8 -	This field is reserved. Reserved
7 RXE	<p>RX Endpoint Enable</p> <p>0 Disabled [Default]</p> <p>1 Enabled</p> <p>An Endpoint should be enabled only after it has been configured.</p>
6 RXR	<p>RX Data Toggle Reset (WS)</p> <p>Write 1 - Reset PID Sequence</p> <p>Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.</p>
5 RXI	<p>RX Data Toggle Inhibit</p> <p>0 Disabled [Default]</p> <p>1 Enabled</p> <p>This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.</p>
4 -	This field is reserved. Reserved.
3–2 RXT	<p>RX Endpoint Type - Read/Write</p> <p>00 Control</p> <p>01 Isochronous</p> <p>10 Bulk</p> <p>11 Reserved</p>
1 RXD	<p>RX Endpoint Data Sink - Read/Write - TBD</p> <p>0 Dual Port Memory Buffer/DMA Engine [Default]</p> <p>Should always be written as zero.</p>
0 RXS	<p>RX Endpoint Stall - Read/Write</p> <p>0 End Point OK. [Default]</p> <p>1 End Point Stalled</p> <p>This bit is set automatically upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It is cleared automatically upon receipt a SETUP request if this Endpoint is configured as a Control Endpoint,</p>

Table continues on the next page...

**USBC\_n\_ENDPTCTRL5 field descriptions (continued)**

Field	Description
	Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues to returning STALL until this bit is either cleared by software or automatically cleared as above.

**52.6.45 Endpoint Control 6 (USBC\_n\_ENDPTCTRL6)**

This is endpoint control register for endpoint 6 in device operation mode.

**NOTE**

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: 218\_4000h base + 1D8h offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**USBC\_n\_ENDPTCTRL6 field descriptions**

Field	Description
31–24 -	This field is reserved. Reserved

Table continues on the next page...



**USBC\_n\_ENDPTCTRL6 field descriptions (continued)**

Field	Description
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It is cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint. Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues to returning STALL until this bit is either cleared by software or automatically cleared as above. <b>NOTE:</b> For CONTROL type endpoint, there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. Take care that the STALL bit is not set immediately after writing a '1' to it. Please follow this procedure: continually write this STALL bit until it is set or until a new setup has ben received by checking the associated ENDPTSETUPSTAT bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 0 Disabled [Default]

*Table continues on the next page...*

**USBC\_n\_ENDPTCTRL6 field descriptions (continued)**

Field	Description
	1 Enabled An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3-2 RXT	RX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Reserved
1 RXD	RX Endpoint Data Sink - Read/Write - TBD 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It is cleared automatically upon receipt a SETUP request if this Endpoint is configured as a Control Endpoint, Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues to returning STALL until this bit is either cleared by software or automatically cleared as above.

**52.6.46 Endpoint Control 7 (USBC\_n\_ENDPTCTRL7)**

This is endpoint control register for endpoint 7 in device operation mode.

**NOTE**

If one endpoint direction is enabled and the paired endpoint of opposite direction is disabled then the unused direction type must be changed from the default control-type to any other type (that is Bulk-type). leaving an unconfigured endpoint control

causes undefined behavior for the data pid tracking on the active endpoint/direction.

Address: 218\_4000h base + 1DCh offset + (512d × i), where i=0d to 1d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								TXE	TXR	TXI	Reserved	TXT		TXD	TXS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								RXE	RXR	RXI	Reserved	RXT		RXD	RXS
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### USBC\_n\_ENDPTCTRL7 field descriptions

Field	Description
31–24 -	This field is reserved. Reserved
23 TXE	TX Endpoint Enable 0 Disabled [Default] 1 Enabled An Endpoint should be enabled only after it has been configured.
22 TXR	TX Data Toggle Reset (WS) Write 1 - Reset PID Sequence Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the Host and device.
21 TXI	TX Data Toggle Inhibit 0 PID Sequencing Enabled. [Default] 1 PID Sequencing Disabled. This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always transmit DATA0 for a data packet.
20 -	This field is reserved. Reserved

Table continues on the next page...

## USBC\_n\_ENDPTCTRL7 field descriptions (continued)

Field	Description
19–18 TXT	TX Endpoint Type - Read/Write 00 Control 01 Isochronous 10 Bulk 11 Interrupt
17 TXD	TX Endpoint Data Source - Read/Write 0 Dual Port Memory Buffer/DMA Engine [DEFAULT] Should always be written as 0.
16 TXS	TX Endpoint Stall - Read/Write 0 End Point OK 1 End Point Stalled  This bit is set automatically upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It is cleared automatically upon receipt of a SETUP request if this Endpoint is configured as a Control Endpoint.  Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues to returning STALL until this bit is either cleared by software or automatically cleared as above.  <b>NOTE:</b> For CONTROL type endpoint, there is a slight delay (50 clocks max) between the ENDPTSETUPSTAT begin cleared and hardware continuing to clear this bit. In most systems, it is unlikely the DCD software will observe this delay. Take care that the STALL bit is not set immediately after writing a '1' to it. Please follow this procedure: continually write this STALL bit until it is set or until a new setup has been received by checking the associated ENDPTSETUPSTAT bit.
15–8 -	This field is reserved. Reserved
7 RXE	RX Endpoint Enable 0 Disabled [Default] 1 Enabled  An Endpoint should be enabled only after it has been configured.
6 RXR	RX Data Toggle Reset (WS) Write 1 - Reset PID Sequence  Whenever a configuration event is received for this Endpoint, software must write a one to this bit in order to synchronize the data PID's between the host and device.
5 RXI	RX Data Toggle Inhibit 0 Disabled [Default] 1 Enabled  This bit is only used for test and should always be written as zero. Writing a one to this bit causes this endpoint to ignore the data toggle sequence and always accept data packet regardless of their data PID.
4 -	This field is reserved. Reserved.
3–2 RXT	RX Endpoint Type - Read/Write 00 Control

Table continues on the next page...

**USBC\_n\_ENDPTCTRL7 field descriptions (continued)**

Field	Description
	01 Isochronous 10 Bulk 11 Reserved
1 RXD	RX Endpoint Data Sink - Read/Write - TBD 0 Dual Port Memory Buffer/DMA Engine [Default] Should always be written as zero.
0 RXS	RX Endpoint Stall - Read/Write 0 End Point OK. [Default] 1 End Point Stalled This bit is set automatically upon receipt of a SETUP request if this Endpoint is not configured as a Control Endpoint. It is cleared automatically upon receipt a SETUP request if this Endpoint is configured as a Control Endpoint, Software can write a one to this bit to force the endpoint to return a STALL handshake to the Host. It continues to returning STALL until this bit is either cleared by software or automatically cleared as above.



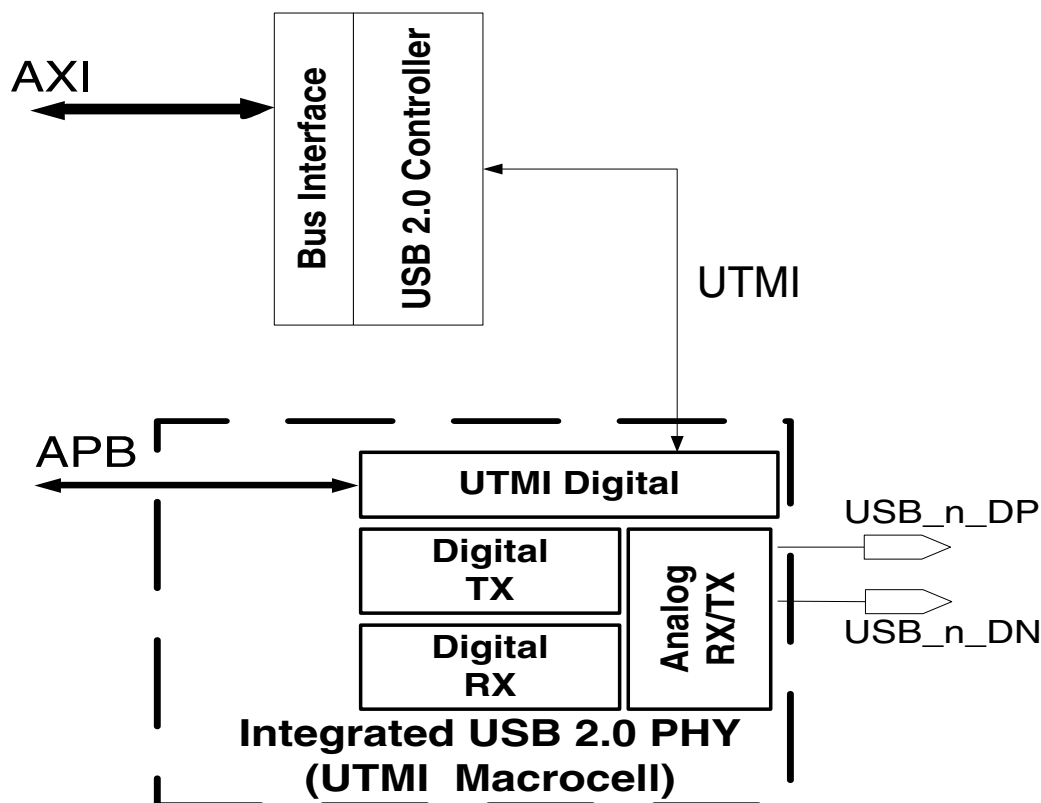
## Chapter 53

# Universal Serial Bus 2.0 Integrated PHY (USB-PHY)

### 53.1 USB PHY Overview

The chip contains 2 integrated USB 2.0 PHY macrocells capable of connecting to USB host/device systems at the USB low-speed (LS) rate of 1.5 Mbits/s, full-speed (FS) rate of 12 Mbits/s or at the USB 2.0 high-speed (HS) rate of 480 Mbits/s.

See [Figure 53-1](#) for a block diagram of the PHY. The integrated PHY provides a standard UTM interface. The USB\_n\_DN and USB\_n\_DP pins connect directly to a USB connector.



**Figure 53-1. USB 2.0 PHY Block Diagram**

USBPHY1 is the PHY interface for USB OTG1 controller; USBPHY2 is the PHY interface for USB OTG2 controller.

The following subsections describe the external interfaces, internal interfaces, major blocks, and programmable registers that comprise the integrated USB 2.0 PHY.

## 53.2 Operation

The UTM provides a 16-bit interface to the USB controller. This interface is clocked at 30 MHz.

- The digital portions of the USBPHY block include the UTMI, digital transmitter, digital receiver, and the programmable registers.
- The analog transceiver section comprises an analog receiver and an analog transmitter, as shown in [Figure 53-2](#).



### 53.2.1 UTMI

The UTMI block handles the line\_state bits, reset buffering, suspend distribution, transceiver speed selection, and transceiver termination selection.

The PLL supplies a 120 MHz signal to all of the digital logic. The UTMI block does a final divide-by-four to develop the 30 MHz clock used in the interface.

### 53.2.2 Digital Transmitter

The digital transmitter receives the 16-bit transmit data from the USB controller and handles the tx\_valid, tx\_validh and tx\_ready handshake.

In addition, it contains the transmit serializer that converts the 16-bit parallel words at 30 MHz to a single bitstream at 480 Mbit for high-speed or 12 Mbit for full-speed or 1.5 Mbit for low-speed. It does this while implementing the bit-stuffing algorithm and the NRZI encoder that are used to remove the DC component from the serial bitstream. The output of this encoder is sent to the low-speed (LS), full-speed (FS) or high-speed (HS) drivers in the analog transceiver section's transmitter block.

### 53.2.3 Digital Receiver

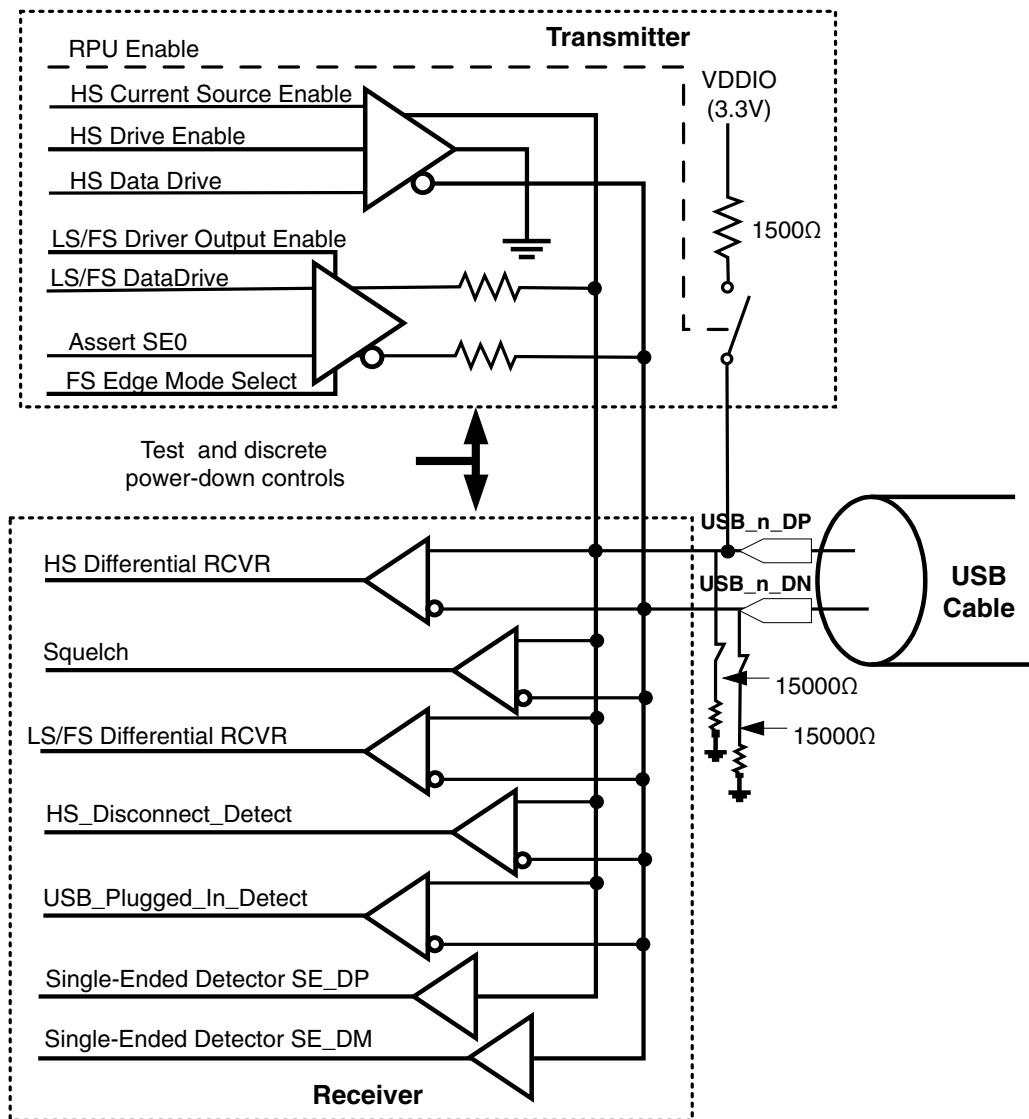
The digital receiver receives the raw serial bitstream from the low speed (LS) differential transceiver, full speed (FS) differential transceiver, and a 9X, 480 MHz sampled data from the high speed (HS) differential transceiver.

As the phase of the USB host transmitter shifts relative to the local PLL, the receiver section's HS DLL tracks these changes to give a reliable sample of the incoming 480 Mbit/s bitstream. Since this sample point shifts relative to the PLL phase used by the digital logic, a rate-matching elastic buffer is provided to cross this clock domain boundary. Once the bitstream is in the local clock domain, an NRZI decoder and bit unstuffer restore the original payload data bitstream and pass it to a deserializer and holding register. The receive state machine handles the rx\_valid, rx\_validh, and handshake with the USB controller. The handshake is not interlocked, in that there is no rx\_ready signal coming from the controller. The controller must take each 16-bit value as presented by the PHY. The receive state machine provides an rx\_active signal to the controller that indicates when it is inside a valid packet (SYNC detected, and so on).

## 53.2.4 Analog Receiver

The analog receiver comprises five differential receivers, two single-ended receivers, and a 9X, 480 MHz HS data sampling module

, as shown in the figure below and described further in this section.



**Figure 53-2. USB 2.0 PHY Analog Transceiver Block Diagram**

### 53.2.4.1 HS Differential Receiver

The high-speed differential receiver is both a differential analog receiver and threshold comparator. Its output is a one if the differential signal is greater than a 0-V threshold.

Otherwise, its output is 0. Its purpose is to discriminate the  $\pm 400$ -mV differential voltage resulting from the high-speed drivers current flow into the dual  $45\Omega$  terminations found on each pin of the differential pair. The envelope or squelch detector, described below, ensures that the differential signal has sufficient magnitude to be valid. The HS differential receiver tolerates up to 500 mV of common mode offset.

#### 53.2.4.2 Squelch Detector

The squelch detector is a differential analog receiver and threshold comparator.

Its output is 1, if the differential magnitude is less than a nominal 100 mV threshold. Otherwise, its output is 0.

Its purpose is to invalidate the HS differential receiver when the incoming signal is simply too low to receive reliably.

#### 53.2.4.3 LS/FS Differential Receiver

The low-speed/full-speed differential receiver is both a differential analog receiver and threshold comparator.

The crossover voltage falls between 1.3 V and 2.0 V. Its output is 1, when the USB<sub>n</sub>\_DP line is above the crossover point and the USB<sub>n</sub>\_DN line is below the crossover point. The digital receiver section decodes the receiver data into J or K state according to the speed.

#### 53.2.4.4 HS Disconnect Detector

It is a differential analog receiver and threshold comparator. It outputs high when differential magnitude is greater than a nominal 575-mV threshold. Otherwise, it outputs low.

#### 53.2.4.5 USB Plugged-In Detector

The USB plugged-in detector looks for both USB<sub>n</sub>\_DP and USB<sub>n</sub>\_DN to be high. There is a pair of large on-chip pullup resistors (200 K $\Omega$ ) that hold both USB<sub>n</sub>\_DP and USB<sub>n</sub>\_DN high when the USB cable is not attached. The USB plugged-in detector signals a 0 in this case.

When operating in device mode, the upstream port in host/hub interface contains a 15 K $\Omega$  pulldown resistor which could easily override the 200 K $\Omega$  pullup resistor. When plugged in, at least one signal in the pair will be low, which will force the plugged-in detector's output high.

#### 53.2.4.6 Single-Ended USB\_DP Receiver

The single-ended USB\_n\_DP receiver output is high whenever the USB\_n\_DP input is above its nominal 1.8 V threshold.

#### 53.2.4.7 Single-Ended USB\_DN Receiver

The single-ended USB\_n\_DN receiver output is high whenever the USB\_n\_DN input is above its nominal 1.8 V threshold.

#### 53.2.4.8 9X Oversample Module

The 9X oversample module uses nine identically spaced phases of the 480 MHz clock to sample a high speed bit data. The squelch signal is sampled only 1X.

### 53.2.5 Analog Transmitter

The analog transmitter comprises two differential drivers: one for high-speed signaling and one for full-speed signaling. It also contains the switchable 1.5 K $\Omega$  pullup resistor.

See [Figure 53-2](#).

#### 53.2.5.1 Switchable High-Speed 45 $\Omega$ Termination Resistors

High-speed current mode differential signaling requires good 90  $\Omega$  differential termination at each end of the USB cable. This results from switching in 45  $\Omega$  terminating resistors from each signal line to ground at each end of the cable.

Because each signal is parallel terminated with 45  $\Omega$  at each end, each driver sees a 22.5  $\Omega$  load. This load impedance is much too low for full-speed signaling levels—hence the need for switchable high-speed terminating resistors. Switchable trimming resistors are provided to tune the actual termination resistance of each device, as shown in [Figure](#)

**53-3.** The HW\_USBPHY\_TX\_TXCAL45DP bit field, for example, allows one of 16 trimming resistor values to be placed in parallel with the 45 $\Omega$  terminator on the USB<sub>n</sub>\_DP signal.

### 53.2.5.2 Low-Speed/Full-Speed Differential Driver

The low-speed/full-speed differential drivers are essentially low-impedance pulldown devices that are switched in a differential mode for low-speed or full-speed signaling, that is, either one or the other device is turned on to signal the "J" state or the "K" state.

### 53.2.5.3 High-Speed Differential Driver

The high-speed differential driver receives a 17.78 mA current from the constant current source ( $I_{ref}$ ) and essentially steers it down either the USB\_DP signal or the USB\_DN signal or alternatively to ground.

This current will produce approximately a 400 mV drop across the 22.5  $\Omega$  termination seen by the driver when it is steered onto one of the signal lines. The approximately 17.78 mA current source is referenced back to the integrated voltage-band-gap ( $V_{bg}$ ) circuit. The  $I_{ref}$ ,  $I_{bias}$ , and  $V$  to  $I$  circuits are shared with the integrated battery charger.

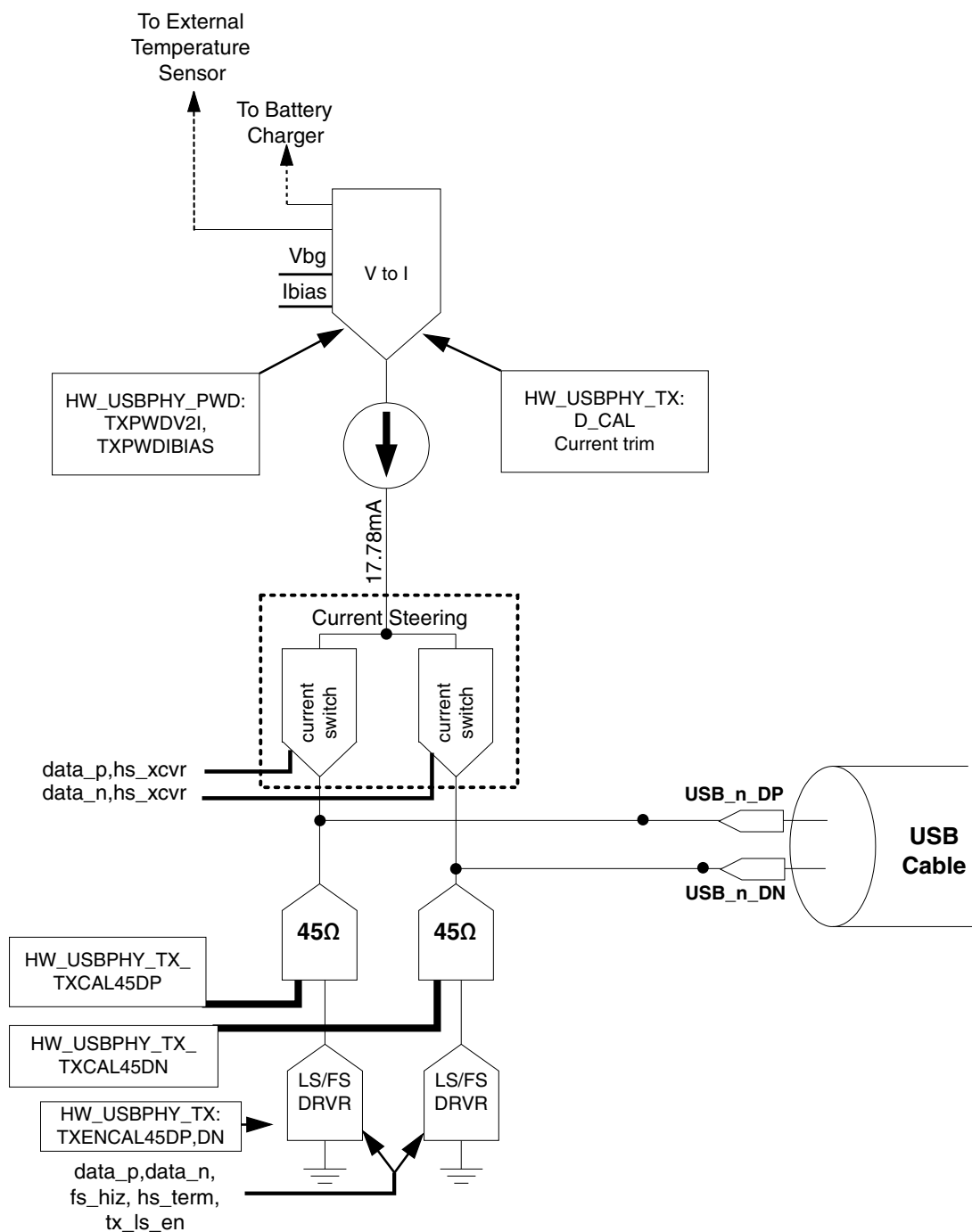
### 53.2.5.4 Switchable 1.5K $\Omega$ USB\_DP Pullup Resistor

This product contains a switchable 1.5 K $\Omega$  pullup resistor on the USB<sub>n</sub>\_DP signal.

This resistor is switched on to indicate to the host/hub controller that a full-speed-capable device is on the USB cable, powered on, and ready. This resistor is switched off at power-on reset so the host does not recognize a USB device until the processor software enables the announcement of a full-speed device.

### 53.2.5.5 Switchable 15K $\Omega$ USB\_DP Pulldown Resistor

This product contains a switchable 15 K $\Omega$  pulldown resistor on both USB\_n\_DP and USB\_n\_DN signals. This is used in host mode to indicate to the device controller that a host is present.



**Figure 53-3. USB 2.0 PHY Transmitter Block Diagram**

## 53.2.6 Recommended Register Configuration for USB Certification

The register settings in this section are recommended for passing USB certification.

The following settings lower the J/K levels to certifiable limits:

```
HW_USBPHY_TX_TXCAL45DP = 0x0
HW_USBPHY_TX_TXCAL45DN = 0x0
HW_USBPHY_TX_D_CAL = 0x7
```

## 53.2.7 Charger detection

The USB charger detector is a block that detects whether the upstream-facing device is connected to a down-stream facing charger, either a dedicated USB charger or a host charger.

The USB charger detector is comprised of two sub-blocks, namely the USB data-pin contact detector and the charger detector.

This section details those two sub-blocks and gives the software flow of USB charger detection. Finally, this chapter discusses the detection of a USB charger in case of a dead battery.

### 53.2.7.1 Charger detect control table

Before we dive into the details of the detectors, we show the logic table of the control signals to give the user an overall picture of the charger detector.

**Table 53-1. Charger detection control table**

EN_B	CHK_CHRG_B	CHK_CONTACT	Data pin contact detector	Charger detector
0	1	1	Enabled	Disabled
0	0	x(don't care)	Disabled	Enabled
1	x	x(don't care)	Disabled	Disabled

### 53.2.7.2 Data pin contact detector

According to Battery Charging Specification (rev 1.2), USB plugs and receptacles are designed such that when the plug is inserted into the receptacle, the power pins make contact before the data pins make contact. Therefore, there is inevitably a time interval during which USB<sub>n</sub>\_VBUS has been observed by the device while the USB<sub>n</sub>\_DP and USB<sub>n</sub>\_DN pins are not still pending for contact. The USB data pin contact detector is designed to give the software an indication of the contact of the data pins.

To enable the USB data pin contact detector, the user should set the CHK\_CONTACT bit of the USB1\_CHRG\_DETECT register to 1 and monitor the PLUG\_CONTACT bit status of the USB1\_CHRG\_DETECT\_STAT register. If PLUG\_CONTACT is 1, then it indicates that the data pins have made good contacts, otherwise the user should continue to wait until this bit is set.

According to Table 1, it should be noted that the data pin contact detector only works when EN\_B=0 and CHK\_CHRG\_B=1, both bits being of the USB1\_CHRG\_DETECT register.

### 53.2.7.3 Charger detector

Once the data pins make contact, the user should enable the charger detector by clearing the CHK\_CHRG\_B bit that is low-active. Then the user should wait for 40ms and then check the status bit of CHRG\_DETECTED in register hw\_anadig\_usb1\_chrg\_det\_stat. CHRG\_DETECTED=1 means that the device is connected to a charger, either a dedicated charger or a charging downstream port (or equivalently called a host charger, or charging host). To further differentiate between a host charger and a dedicated charger, the user is suggested to pull up USB<sub>n</sub>\_DP signaling a connect event to the host. Then the user should monitor the USB<sub>n</sub>\_DN line status. If USB<sub>n</sub>\_DN=1, then the charger is a dedicated charger; if USB<sub>n</sub>\_DN=0, then it is a host charger.

### 53.2.7.4 Charger detection software flow

Upon seeing VBUS, the software should follow the software flow for the charger detection process. The flow chart mentions the "enable the vdd3p0 current limiter". Please refer to the power chapter for details.



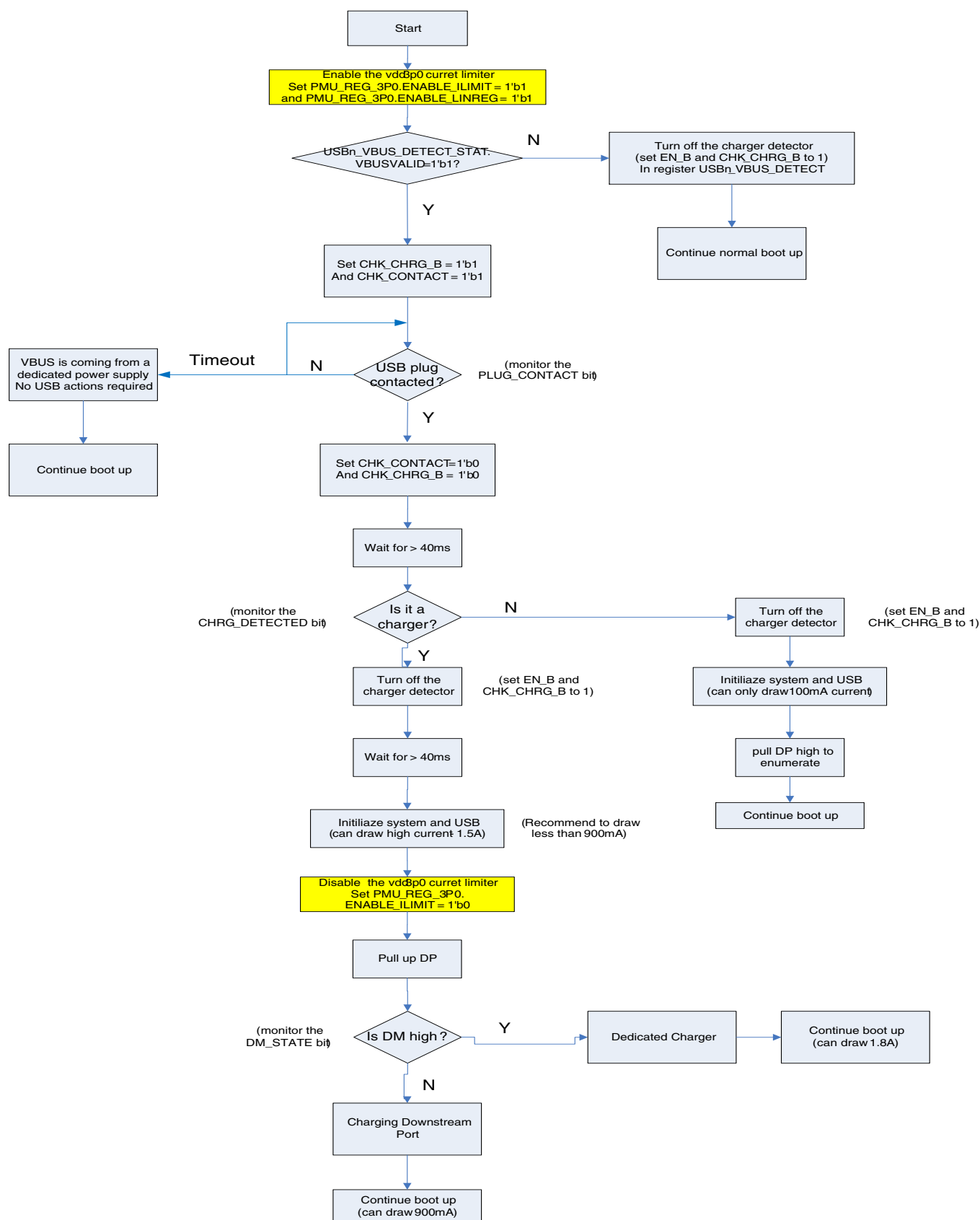


Figure 53-4. USBPHY Charger Detection Software Flow

### 53.2.7.5 Dead Battery Protect

All the descriptions above are based on the assumption that all the power supplies have been on when the device is plugged into a remote host (or charger). However, there are cases when the local battery of the portable device has been so depleted that the system could not be turned on. In such scenarios the user may prefer a method of signaling the external power management unit (PMIC) the existence of the USB charger to draw a current larger than 100mA from the remote host to speed up system boot up or battery charging. The charger detector indeed supports this function.

When we have a fully depleted battery, all the power supplies might be off. Upon insertion of the 5V, the supplies are brought up by the external PMIC and the internal regulators. Due to the 100mA inrush current limit of the USB spec, we cannot draw larger than 100mA current which might be a limit for system boot-up. Since by default, EN\_B=0, CHK\_CHRG\_B=0 and CHK\_CONTACT=1, the usb charger detector is automatically enabled without any software operation needed and it can signal the external PMIC the existence of a USB charger through the open-drain output pin USB\_OTG\_CHD\_B. This pin should be pulled up to an external voltage that is acceptable to the PMIC. If this signal is low, then the PMIC can get that the device is connected to a charger. In this case, the PMIC can draw more than 100mA current from the USB.

It should be noted that this function requires cooperation between the chip and the external PMIC. It is suggested that the user consult Freescale for such use cases.

## 53.3 USB PHY Memory Map/Register Definition

### USBPHY Hardware Register Format Summary

**USBPHY memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20C_9000	USB PHY Power-Down Register (USBPHY1_PWD)	32	R/W	001E_1C00h	<a href="#">53.3.1/3281</a>
20C_9004	USB PHY Power-Down Register (USBPHY1_PWD_SET)	32	R/W	001E_1C00h	<a href="#">53.3.1/3281</a>
20C_9008	USB PHY Power-Down Register (USBPHY1_PWD_CLR)	32	R/W	001E_1C00h	<a href="#">53.3.1/3281</a>
20C_900C	USB PHY Power-Down Register (USBPHY1_PWD_TOG)	32	R/W	001E_1C00h	<a href="#">53.3.1/3281</a>
20C_9010	USB PHY Transmitter Control Register (USBPHY1_TX)	32	R/W	1006_0607h	<a href="#">53.3.2/3283</a>

*Table continues on the next page...*

## USBPHY memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20C_9014	USB PHY Transmitter Control Register (USBPHY1_TX_SET)	32	R/W	1006_0607h	<a href="#">53.3.2/3283</a>
20C_9018	USB PHY Transmitter Control Register (USBPHY1_TX_CLR)	32	R/W	1006_0607h	<a href="#">53.3.2/3283</a>
20C_901C	USB PHY Transmitter Control Register (USBPHY1_TX_TOG)	32	R/W	1006_0607h	<a href="#">53.3.2/3283</a>
20C_9020	USB PHY Receiver Control Register (USBPHY1_RX)	32	R/W	0000_0000h	<a href="#">53.3.3/3284</a>
20C_9024	USB PHY Receiver Control Register (USBPHY1_RX_SET)	32	R/W	0000_0000h	<a href="#">53.3.3/3284</a>
20C_9028	USB PHY Receiver Control Register (USBPHY1_RX_CLR)	32	R/W	0000_0000h	<a href="#">53.3.3/3284</a>
20C_902C	USB PHY Receiver Control Register (USBPHY1_RX_TOG)	32	R/W	0000_0000h	<a href="#">53.3.3/3284</a>
20C_9030	USB PHY General Control Register (USBPHY1_CTRL)	32	R/W	C020_0000h	<a href="#">53.3.4/3286</a>
20C_9034	USB PHY General Control Register (USBPHY1_CTRL_SET)	32	R/W	C020_0000h	<a href="#">53.3.4/3286</a>
20C_9038	USB PHY General Control Register (USBPHY1_CTRL_CLR)	32	R/W	C020_0000h	<a href="#">53.3.4/3286</a>
20C_903C	USB PHY General Control Register (USBPHY1_CTRL_TOG)	32	R/W	C020_0000h	<a href="#">53.3.4/3286</a>
20C_9040	USB PHY Status Register (USBPHY1_STATUS)	32	R/W	0000_0000h	<a href="#">53.3.5/3289</a>
20C_9050	USB PHY Debug Register (USBPHY1_DEBUG)	32	R/W	7F18_0000h	<a href="#">53.3.6/3291</a>
20C_9054	USB PHY Debug Register (USBPHY1_DEBUG_SET)	32	R/W	7F18_0000h	<a href="#">53.3.6/3291</a>
20C_9058	USB PHY Debug Register (USBPHY1_DEBUG_CLR)	32	R/W	7F18_0000h	<a href="#">53.3.6/3291</a>
20C_905C	USB PHY Debug Register (USBPHY1_DEBUG_TOG)	32	R/W	7F18_0000h	<a href="#">53.3.6/3291</a>
20C_9060	UTMI Debug Status Register 0 (USBPHY1_DEBUG0_STATUS)	32	R	0000_0000h	<a href="#">53.3.7/3293</a>
20C_9070	UTMI Debug Status Register 1 (USBPHY1_DEBUG1)	32	R/W	0000_1000h	<a href="#">53.3.8/3294</a>
20C_9074	UTMI Debug Status Register 1 (USBPHY1_DEBUG1_SET)	32	R/W	0000_1000h	<a href="#">53.3.8/3294</a>
20C_9078	UTMI Debug Status Register 1 (USBPHY1_DEBUG1_CLR)	32	R/W	0000_1000h	<a href="#">53.3.8/3294</a>
20C_907C	UTMI Debug Status Register 1 (USBPHY1_DEBUG1_TOG)	32	R/W	0000_1000h	<a href="#">53.3.8/3294</a>
20C_9080	UTMI RTL Version (USBPHY1_VERSION)	32	R	0402_0000h	<a href="#">53.3.9/3295</a>
20C_A000	USB PHY Power-Down Register (USBPHY2_PWD)	32	R/W	001E_1C00h	<a href="#">53.3.1/3281</a>
20C_A004	USB PHY Power-Down Register (USBPHY2_PWD_SET)	32	R/W	001E_1C00h	<a href="#">53.3.1/3281</a>
20C_A008	USB PHY Power-Down Register (USBPHY2_PWD_CLR)	32	R/W	001E_1C00h	<a href="#">53.3.1/3281</a>
20C_A00C	USB PHY Power-Down Register (USBPHY2_PWD_TOG)	32	R/W	001E_1C00h	<a href="#">53.3.1/3281</a>
20C_A010	USB PHY Transmitter Control Register (USBPHY2_TX)	32	R/W	1006_0607h	<a href="#">53.3.2/3283</a>
20C_A014	USB PHY Transmitter Control Register (USBPHY2_TX_SET)	32	R/W	1006_0607h	<a href="#">53.3.2/3283</a>
20C_A018	USB PHY Transmitter Control Register (USBPHY2_TX_CLR)	32	R/W	1006_0607h	<a href="#">53.3.2/3283</a>
20C_A01C	USB PHY Transmitter Control Register (USBPHY2_TX_TOG)	32	R/W	1006_0607h	<a href="#">53.3.2/3283</a>
20C_A020	USB PHY Receiver Control Register (USBPHY2_RX)	32	R/W	0000_0000h	<a href="#">53.3.3/3284</a>

Table continues on the next page...

## USBPHY memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20C_A024	USB PHY Receiver Control Register (USBPHY2_RX_SET)	32	R/W	0000_0000h	<a href="#">53.3.3/3284</a>
20C_A028	USB PHY Receiver Control Register (USBPHY2_RX_CLR)	32	R/W	0000_0000h	<a href="#">53.3.3/3284</a>
20C_A02C	USB PHY Receiver Control Register (USBPHY2_RX_TOG)	32	R/W	0000_0000h	<a href="#">53.3.3/3284</a>
20C_A030	USB PHY General Control Register (USBPHY2_CTRL)	32	R/W	C020_0000h	<a href="#">53.3.4/3286</a>
20C_A034	USB PHY General Control Register (USBPHY2_CTRL_SET)	32	R/W	C020_0000h	<a href="#">53.3.4/3286</a>
20C_A038	USB PHY General Control Register (USBPHY2_CTRL_CLR)	32	R/W	C020_0000h	<a href="#">53.3.4/3286</a>
20C_A03C	USB PHY General Control Register (USBPHY2_CTRL_TOG)	32	R/W	C020_0000h	<a href="#">53.3.4/3286</a>
20C_A040	USB PHY Status Register (USBPHY2_STATUS)	32	R/W	0000_0000h	<a href="#">53.3.5/3289</a>
20C_A050	USB PHY Debug Register (USBPHY2_DEBUG)	32	R/W	7F18_0000h	<a href="#">53.3.6/3291</a>
20C_A054	USB PHY Debug Register (USBPHY2_DEBUG_SET)	32	R/W	7F18_0000h	<a href="#">53.3.6/3291</a>
20C_A058	USB PHY Debug Register (USBPHY2_DEBUG_CLR)	32	R/W	7F18_0000h	<a href="#">53.3.6/3291</a>
20C_A05C	USB PHY Debug Register (USBPHY2_DEBUG_TOG)	32	R/W	7F18_0000h	<a href="#">53.3.6/3291</a>
20C_A060	UTMI Debug Status Register 0 (USBPHY2_DEBUG0_STATUS)	32	R	0000_0000h	<a href="#">53.3.7/3293</a>
20C_A070	UTMI Debug Status Register 1 (USBPHY2_DEBUG1)	32	R/W	0000_1000h	<a href="#">53.3.8/3294</a>
20C_A074	UTMI Debug Status Register 1 (USBPHY2_DEBUG1_SET)	32	R/W	0000_1000h	<a href="#">53.3.8/3294</a>
20C_A078	UTMI Debug Status Register 1 (USBPHY2_DEBUG1_CLR)	32	R/W	0000_1000h	<a href="#">53.3.8/3294</a>
20C_A07C	UTMI Debug Status Register 1 (USBPHY2_DEBUG1_TOG)	32	R/W	0000_1000h	<a href="#">53.3.8/3294</a>
20C_A080	UTMI RTL Version (USBPHY2_VERSION)	32	R	0402_0000h	<a href="#">53.3.9/3295</a>

### 53.3.1 USB PHY Power-Down Register (USBPHYx\_PWDn)

The USB PHY Power-Down Register provides overall control of the PHY power state. Before programming this register, the PHY clocks must be enabled in registers USBPHYx\_CTRLn and CCM\_ANALOG\_USBPHYx\_PLL\_480\_CTRLn.

Address: Base address + 0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	RSVD2												RXPWDRX	RXPWDDIFF	RXPWD1PT1	RXPWDENV	RSVD1
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	0	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD1			TXPWDV2I	TXPWDIBIAS	TXPWDFS	RSVD0									
W																
Reset	0	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0

**USBPHYx\_PWDn field descriptions**

Field	Description
31–21 RSVD2	Reserved.
20 RXPWDRX	0 = Normal operation. 1 = Power-down the entire USB PHY receiver block except for the full-speed differential receiver. Note that this bit will be auto cleared if there is USB wakeup event while ENAUTOCLR_PHY_PWD bit of USBPHYx_CTRL is enabled.
19 RXPWDDIFF	0 = Normal operation. 1 = Power-down the USB high-speed differential receiver.

*Table continues on the next page...*

## USBPHYx\_PWDn field descriptions (continued)

Field	Description
	Note that this bit will be auto cleared if there is USB wakeup event while ENAUTOCLR_PHY_PWD bit of USBPHYx_CTRL is enabled.
18 RXPWD1PT1	<p>0 = Normal operation.</p> <p>1 = Power-down the USB full-speed differential receiver.</p> <p>Note that this bit will be auto cleared if there is USB wakeup event while ENAUTOCLR_PHY_PWD bit of USBPHYx_CTRL is enabled.</p>
17 RXPWDENV	<p>0 = Normal operation.</p> <p>1 = Power-down the USB high-speed receiver envelope detector (squelch signal).</p> <p>Note that this bit will be auto cleared if there is USB wakeup event while ENAUTOCLR_PHY_PWD bit of USBPHYx_CTRL is enabled.</p>
16–13 RSVD1	Reserved.
12 TXPWDV2I	<p>0 = Normal operation.</p> <p>1 = Power-down the USB PHY transmit V-to-I converter and the current mirror.</p> <p>Note that this bit will be auto cleared if there is USB wakeup event while ENAUTOCLR_PHY_PWD bit of USBPHYx_CTRL is enabled.</p> <p>Note that these circuits are shared with the battery charge circuit. Setting this to 1 does not power-down these circuits, unless the corresponding bit in the battery charger is also set for power-down.</p>
11 TXPWDIBIAS	<p>0 = Normal operation.</p> <p>1 = Power-down the USB PHY current bias block for the transmitter. This bit should be set only when the USB is in suspend mode. This effectively powers down the entire USB transmit path.</p> <p>Note that this bit will be auto cleared if there is USB wakeup event while ENAUTOCLR_PHY_PWD bit of USBPHYx_CTRL is enabled.</p> <p>Note that these circuits are shared with the battery charge circuit. Setting this bit to 1 does not power-down these circuits, unless the corresponding bit in the battery charger is also set for power-down.</p>
10 TXPWDFS	<p>0 = Normal operation.</p> <p>1 = Power-down the USB full-speed drivers. This turns off the current starvation sources and puts the drivers into high-impedance output.</p> <p>Note that this bit will be auto cleared if there is USB wakeup event while ENAUTOCLR_PHY_PWD bit of USBPHYx_CTRL is enabled.</p>
9–0 RSVD0	Reserved.

## 53.3.2 USB PHY Transmitter Control Register (USBPHYx\_TXn)

The USB PHY Transmitter Control Register handles the transmit controls.

Address: Base address + 10h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RSVD5			USBPHY_TX_EDGECTRL			RSVD2				TXCAL45DP					
W																
Reset	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD1				TXCAL45DN				RSVD0				D_CAL			
W																
Reset	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1	1

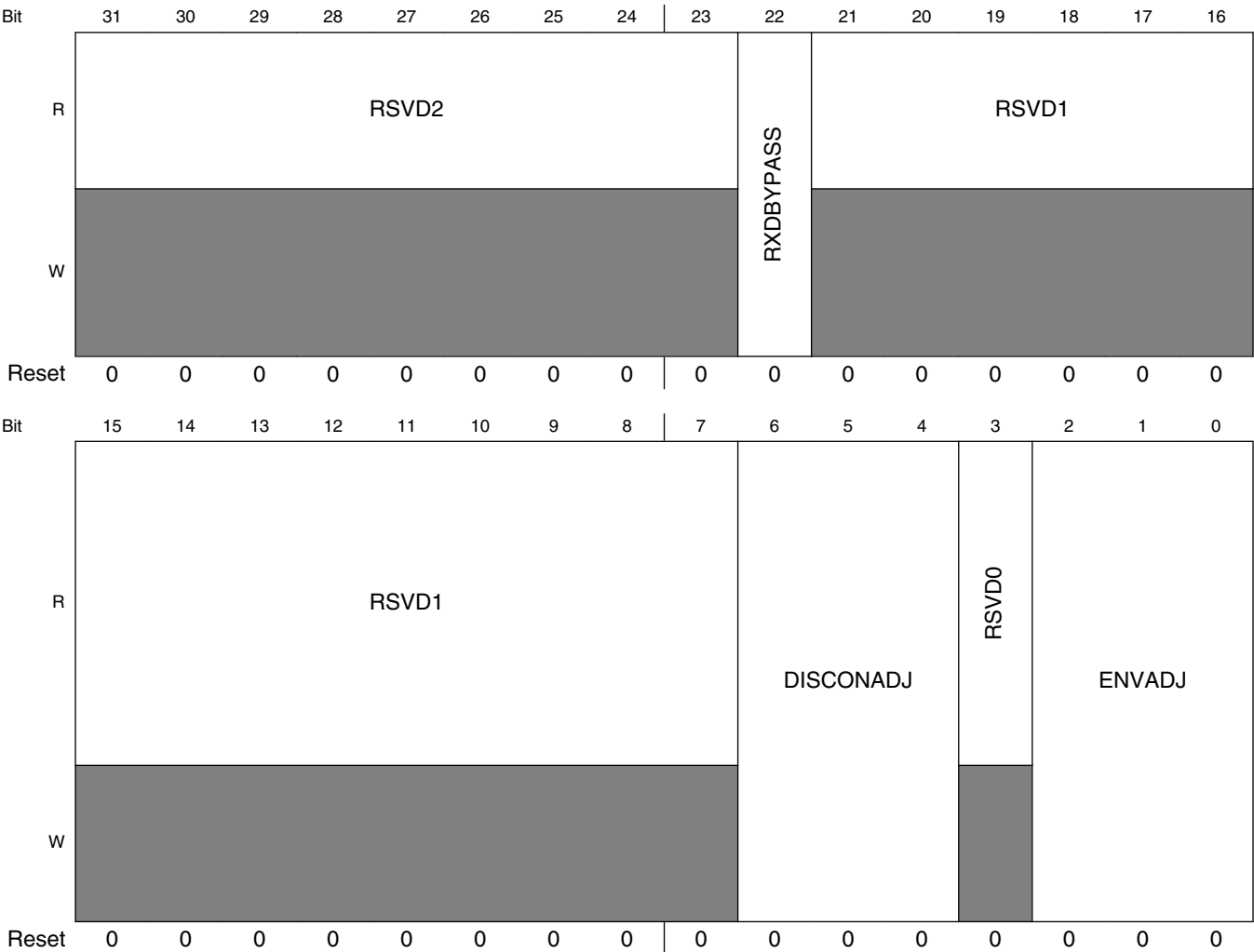
### USBPHYx\_TXn field descriptions

Field	Description
31–29 RSVD5	Reserved.
28–26 USBPHY_TX_EDGECTRL	Controls the edge-rate of the current sensing transistors used in HS transmit. NOT FOR CUSTOMER USE.
25–20 RSVD2	Reserved.
19–16 TXCAL45DP	Decode to select a 45-Ohm resistance to the USB_DP output pin. Maximum resistance = 0000. Resistance is centered by design at 0110.
15–12 RSVD1	Reserved. <b>Note:</b> This bit should remain clear.
11–8 TXCAL45DN	Decode to select a 45-Ohm resistance to the USB_DN output pin. Maximum resistance = 0000. Resistance is centered by design at 0110.
7–4 RSVD0	Reserved. <b>Note:</b> This bit should remain clear.
3–0 D_CAL	Resistor Trimming Code: 0000 = 0.16% 0111 = Nominal 1111 = +25%

### 53.3.3 USB PHY Receiver Control Register (USBPHYx\_RXn)

The USB PHY Receiver Control Register handles receive path controls.

Address: Base address + 20h offset + (4d × i), where i=0d to 3d



USBPHYx\_RXn field descriptions

Field	Description
31–23 RSVD2	Reserved.
22 RXDBYPASS	0 = Normal operation. 1 = Use the output of the USB_DP single-ended receiver in place of the full-speed differential receiver. This test mode is intended for lab use only.
21–7 RSVD1	Reserved.

Table continues on the next page...



**USBPHYx\_RXn field descriptions (continued)**

<b>Field</b>	<b>Description</b>
6–4 DISCONADJ	<p>The DISCONADJ field adjusts the trip point for the disconnect detector:</p> <p>000 = Trip-Level Voltage is 0.57500 V</p> <p>001 = Trip-Level Voltage is 0.56875 V</p> <p>010 = Trip-Level Voltage is 0.58125 V</p> <p>011 = Trip-Level Voltage is 0.58750 V</p> <p>1XX = Reserved</p>
3 RSVD0	Reserved.
2–0 ENVADJ	<p>The ENVADJ field adjusts the trip point for the envelope detector.</p> <p>000 = Trip-Level Voltage is 0.12500 V</p> <p>001 = Trip-Level Voltage is 0.10000 V</p> <p>010 = Trip-Level Voltage is 0.13750 V</p> <p>011 = Trip-Level Voltage is 0.15000 V</p> <p>1XX = Reserved</p>

### 53.3.4 USB PHY General Control Register (USBPHYx\_CTRLn)

The USB PHY General Control Register handles OTG and Host controls. This register also includes interrupt enables and connectivity detect enables and results.

Address: Base address + 30h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	SFTRST	CLKGATE	UTMI_SUSPENDM	HOST_FORCE_LS_SE0	OTG_ID_VALUE	RSVD1		FSDLL_RST_EN	ENVBUSCHG_WKUP	ENIDCHG_WKUP	ENDPDMCHG_WKUP	ENAUTOCLR_PHY_PWD	ENAUTOCLR_CLKGATE	RSVD0	WAKEUP_IRQ	ENIRQWAKEUP
W																
Reset	1	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	ENUTMILEVEL3	ENUTMILEVEL2	DATA_ON_LRADC	DEVPLUGIN_IRQ	ENIRQDEVPLUGIN	RESUME_IRQ	ENIRQRESUMEDETECT	RESUMEIRQSTICKY	ENOTGIDDETECT	OTG_ID_CHG_IRQ	DEVPLUGIN_POLARITY	ENDEVPLUGINDETECT	HOSTDISCONDETECT_IRQ	ENIRQHOSTDISCON	ENHOSTDISCONDETECT	ENOTG_ID_CHG_IRQ
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**USBPHYx\_CTRLn field descriptions**

Field	Description
31 SFTRST	Writing a 1 to this bit will soft-reset the USBPHYx_PWD, USBPHYx_TX, USBPHYx_RX, and USBPHYx_CTRL registers. Set to 0 to release the PHY from reset.
30 CLKGATE	Gate UTMI Clocks. Clear to 0 to run clocks. Set to 1 to gate clocks. Set this to save power while the USB is not actively being used. Configuration state is kept while the clock is gated.  Note this bit can be auto-cleared if there is any wakeup event when USB is suspended while ENAUTOCLR_CLKGATE bit of USBPHYx_CTRL is enabled.
29 UTMI_SUSPENDM	Used by the PHY to indicate a powered-down state. If all the power-down bits in the USBPHYx_PWD are enabled, UTMI_SUSPENDM will be 0, otherwise 1. UTMI_SUSPENDM is negative logic, as required by the UTMI specification.
28 HOST_FORCE_LS_SE0	Forces the next FS packet that is transmitted to have a EOP with LS timing. This bit is used in host mode for the resume sequence. After the packet is transferred, this bit is cleared. The design can use this function to force the LS SE0 or use the USBPHYx_CTRL_UTMI_SUSPENDM to trigger this event when leaving suspend. This bit is used in conjunction with USBPHYx_DEBUG_HOST_RESUME_DEBUG.
27 OTG_ID_VALUE	Almost same as OTGID_STATUS in USBPHYx_STATUS Register. The only difference is that OTG_ID_VALUE has debounce logic to filter the glitches on ID Pad.
26–25 RSVD1	Reserved.
24 FSDLL_RST_EN	Enables the feature to reset the FSDLL lock detection logic at the end of each TX packet.
23 ENVBUSCHG_WKUP	Enables the feature to wakeup USB if VBUS is toggled when USB is suspended.
22 ENIDCHG_WKUP	Enables the feature to wakeup USB if ID is toggled when USB is suspended.
21 ENDPDMCHG_WKUP	Enables the feature to wakeup USB if DP/DM is toggled when USB is suspended. This bit is enabled by default.
20 ENAUTOCLR_PHY_PWD	Enables the feature to auto-clear the PWD register bits in USBPHYx_PWD if there is wakeup event while USB is suspended. This should be enabled if needs to support auto wakeup without S/W's interaction.
19 ENAUTOCLR_CLKGATE	Enables the feature to auto-clear the CLKGATE bit if there is wakeup event while USB is suspended. This should be enabled if needs to support auto wakeup without S/W's interaction.
18 RSVD0	Reserved.
17 WAKEUP_IRQ	Indicates that there is a wakeup event. Reset this bit by writing a 1 to the clear address space and not by a general write.
16 ENIRQWAKEUP	Enables interrupt for the wakeup events.
15 ENUTMILEVEL3	Enables UTMI+ Level3. This should be enabled if needs to support external FS Hub with LS device connected
14 ENUTMILEVEL2	Enables UTMI+ Level2. This should be enabled if needs to support LS device
13 DATA_ON_LRADC	Enables the LRADC to monitor USB_DP and USB_DM. This is for use in non-USB modes only.
12 DEVPLUGIN_IRQ	Indicates that the device is connected. Reset this bit by writing a 1 to the clear address space and not by a general write.
11 ENIRQDEVPLUGIN	Enables interrupt for the detection of connectivity to the USB line.

*Table continues on the next page...*

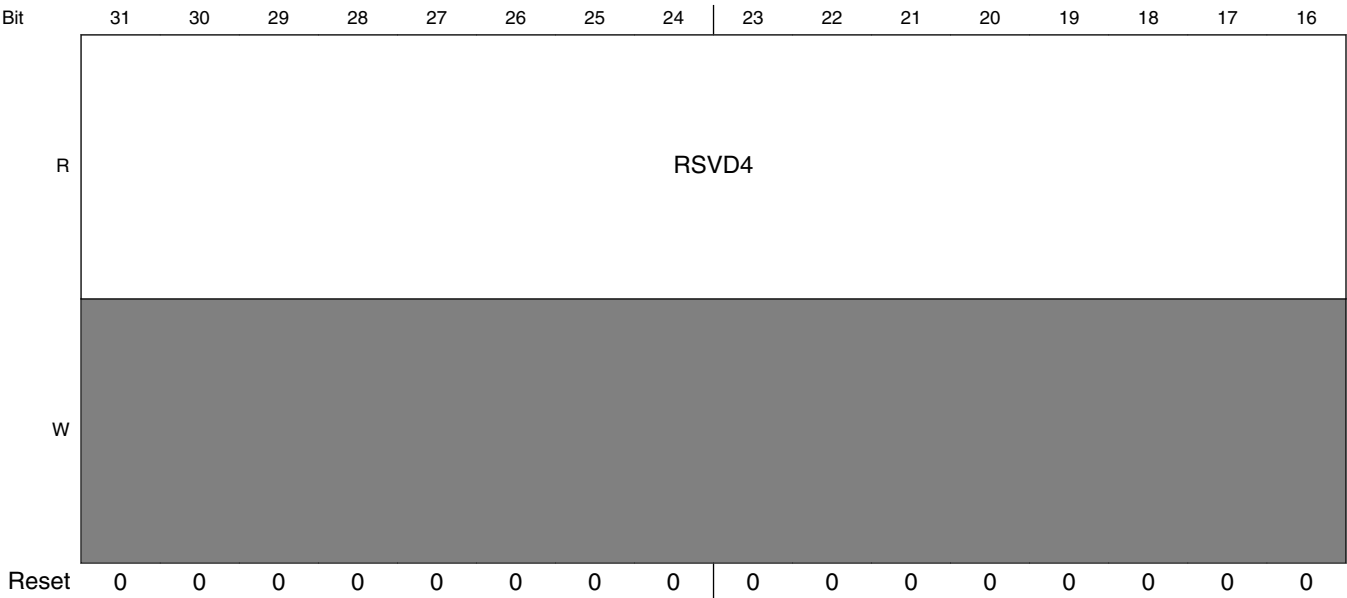
**USBPHYx\_CTRLn field descriptions (continued)**

Field	Description
10 RESUME_IRQ	Indicates that the host is sending a wake-up after suspend. This bit is also set on a reset during suspend. Use this bit to wake up from suspend for either the resume or the reset case. Reset this bit by writing a 1 to the clear address space and not by a general write.
9 ENIRQRESUMEDetect	Enables interrupt for detection of a non-J state on the USB line. This should only be enabled after the device has entered suspend mode.
8 RESUMEIRQSTICKY	Set to 1 will make RESUME_IRQ bit a sticky bit until software clear it. Set to 0, RESUME_IRQ only set during the wake-up period.
7 ENOTGIDDETECT	Enables circuit to detect resistance of MiniAB ID pin.
6 OTG_ID_CHG_IRQ	OTG ID change interrupt. Indicates the value of ID pin changed.
5 DEVPLUGIN_POLARITY	For device mode, if this bit is cleared to 0, then it trips the interrupt if the device is plugged in. If set to 1, then it trips the interrupt if the device is unplugged.
4 ENDEVPLUGINDETECT	For device mode, enables 200-KOhm pullups for detecting connectivity to the host.
3 HOSTDISCONDETECT_IRQ	Indicates that the device has disconnected in high-speed mode. Reset this bit by writing a 1 to the clear address space and not by a general write.
2 ENIRQHOSTDISCON	Enables interrupt for detection of disconnection to Device when in high-speed host mode. This should be enabled after ENDEVPLUGINDETECT is enabled.
1 ENHOSTDISCONDETECT	For host mode, enables high-speed disconnect detector. This signal allows the override of enabling the detection that is normally done in the UTMI controller. The UTMI controller enables this circuit whenever the host sends a start-of-frame packet.  SW shall set this bit when it found the high-speed device is connected, suggested during bus reset, after found high-speed device in USB_PORTSC1.PSPD).  SW shall make sure this bit is not set at the end of resume, otherwise a wrong disconnect status may be detected. Suggest clear it after set USB_PORTSC1.SUSP, set it again after resume is ended(USB_PORTSC1.FPR==0).
0 ENOTG_ID_CHG_IRQ	Enable OTG_ID_CHG_IRQ.

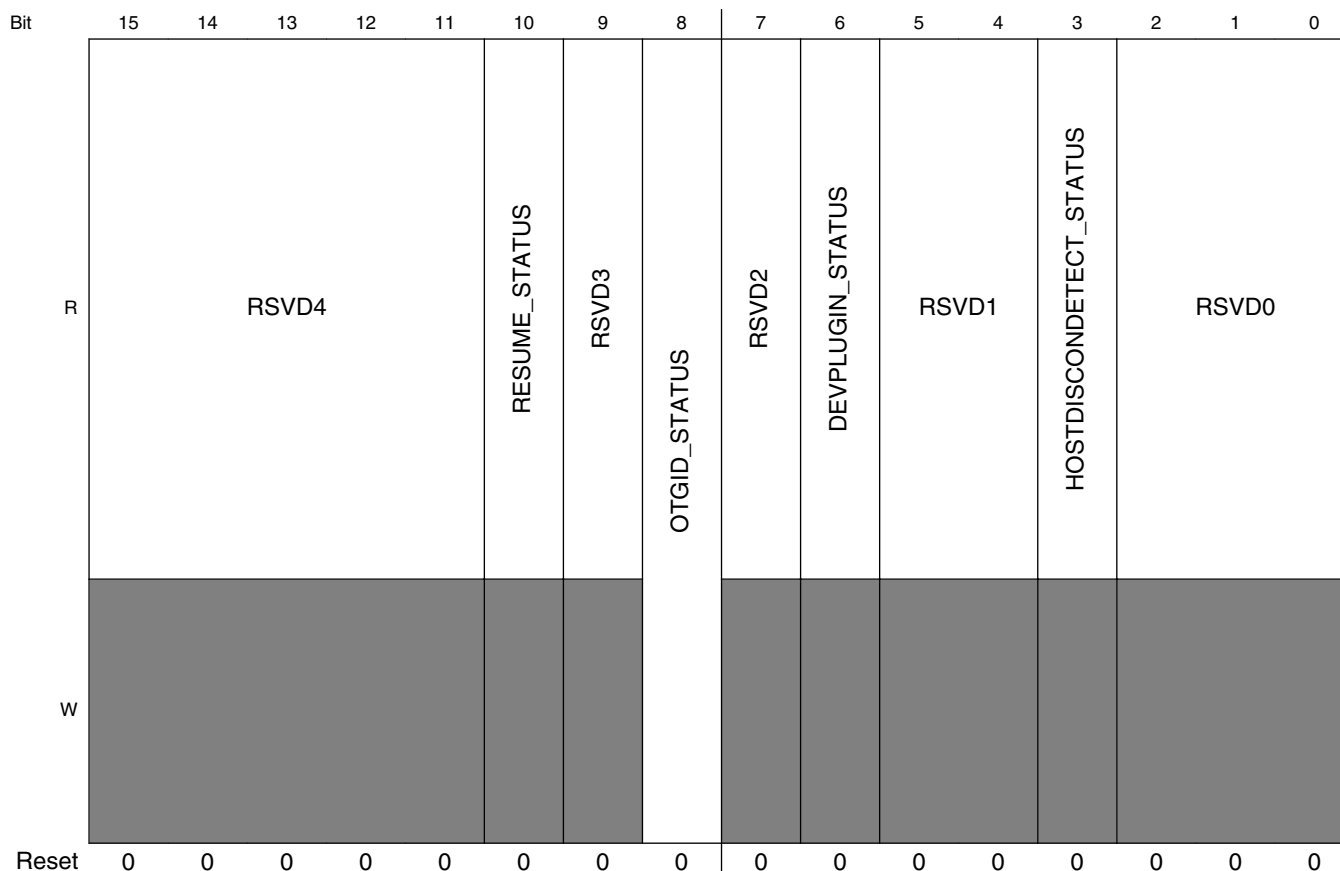
### 53.3.6 USB PHY Status Register (USBPHYx\_STATUS)

The USB PHY Status Register holds results of IRQ and other detects.

Address: Base address + 40h offset



## USB PHY Memory Map/Register Definition



### USBPHYx\_STATUS field descriptions

Field	Description
31–11 RSVD4	Reserved.
10 RESUME_STATUS	Indicates that the host is sending a wake-up after suspend and has triggered an interrupt.
9 RSVD3	Reserved.
8 OTGID_STATUS	Indicates the results of ID pin on MiniAB plug. False (0) is when ID resistance is less than Ra_Plug_ID, indicating host (A) side. True (1) is when ID resistance is greater than Rb_Plug_ID, indicating device (B) side.
7 RSVD2	Reserved.
6 DEVPLUGIN_STATUS	Indicates that the device has been connected on the USB_DP and USB_DM lines.
5–4 RSVD1	Reserved.
3 HOSTDISCONDETECT_STATUS	Indicates that the device has disconnected while in high-speed host mode.
2–0 RSVD0	Reserved.

### 53.3.6 USB PHY Debug Register (USBPHYx\_DEBUGn)

This register is used to debug the USB PHY.

Address: Base address + 50h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	RSVD3		CLKGATE	HOST_RESUME_DEBUG	SQUELCHRESETLENGTH				ENSQUELCHRESET	RSVD2			SQUELCHRESETCOUNT			
W																
Reset	0	1	1	1	1	1	1	1	0	0	0	1	1	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	RSVD1			ENTX2RXCOUNT	TX2RXCOUNT				RSVD0		ENHSTPULLDOWN	HSTPULLDOWN	DEBUG_INTERFACE_HOLD	OTGIDPIOLOCK		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**USBPHYx\_DEBUGn field descriptions**

Field	Description
31 RSVD3	Reserved.
30 CLKGATE	Gate Test Clocks. Clear to 0 for running clocks. Set to 1 to gate clocks. Set this to save power while the USB is not actively being used. Configuration state is kept while the clock is gated.
29 HOST_RESUME_DEBUG	Choose to trigger the host resume SE0 with HOST_FORCE_LS_SE0 = 0 or UTMI_SUSPEND = 1.
28–25 SQUELCHRESETLENGTH	Duration of RESET in terms of the number of 480-MHz cycles.
24 ENSQUELCHRESET	Set bit to allow squelch to reset high-speed receive.
23–21 RSVD2	Reserved.
20–16 SQUELCHRESETCOUNT	Delay in between the detection of squelch to the reset of high-speed RX.
15–13 RSVD1	Reserved.
12 ENTX2RXCOUNT	Set this bit to allow a countdown to transition in between TX and RX.
11–8 TX2RXCOUNT	Delay in between the end of transmit to the beginning of receive. This is a Johnson count value and thus will count to 8.
7–6 RSVD0	Reserved.
5–4 ENHSTPULLDOWN	Set bit 5 to 1 to override the control of the USB_DP 15-KOhm pulldown. Set bit 4 to 1 to override the control of the USB_DM 15-KOhm pulldown. Clear to 0 to disable.
3–2 HSTPULLDOWN	Set bit 3 to 1 to pull down 15-KOhm on USB_DP line. Set bit 2 to 1 to pull down 15-KOhm on USB_DM line. Clear to 0 to disable.
1 DEBUG_INTERFACE_HOLD	Use holding registers to assist in timing for external UTMI interface.
0 OTGIDPIOLOCK	Once OTG ID from USBPHYx_STATUS_OTGID_STATUS, use this to hold the value. This is to save power for the comparators that are used to determine the ID status.



### 53.3.8 UTMI Debug Status Register 0 (USBPHYx\_DEBUG0\_STATUS)

The UTMI Debug Status Register 0 holds multiple views for counters and status of state machines. This is used in conjunction with the USBPHYx\_DEBUG1\_DBG\_ADDRESS field to choose which function to view. The default is described in the bit fields below and is used to count errors.

Address: Base address + 60h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SQUELCH_COUNT						UTMI_RXERROR_FAIL_COUNT										LOOP_BACK_FAIL_COUNT															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

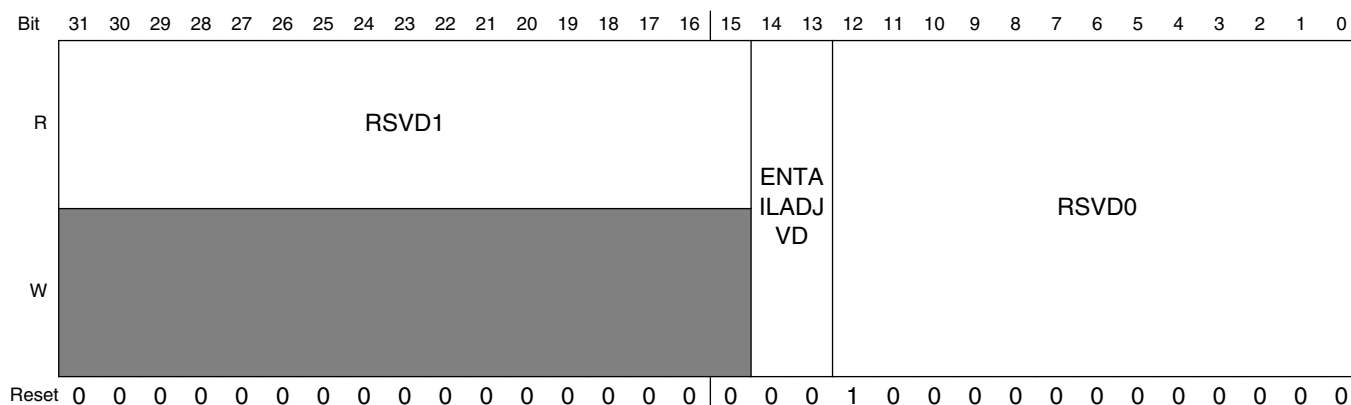
#### USBPHYx\_DEBUG0\_STATUS field descriptions

Field	Description
31–26 SQUELCH_COUNT	Running count of the squelch reset instead of normal end for HS RX.
25–16 UTMI_RXERROR_FAIL_COUNT	Running count of the UTMI_RXERROR.
15–0 LOOP_BACK_FAIL_COUNT	Running count of the failed pseudo-random generator loopback. Each time entering testmode, counter goes to 900D and will count up for every detected packet failure in digital/analog loopback tests.

### 53.3.8 UTMI Debug Status Register 1 (USBPHYx\_DEBUG1n)

Chooses the muxing of the debug register to be shown in USBPHY<sub>X</sub> DEBUG0 STATUS.

Address: Base address + 70h offset + (4d × i), where i=0d to 3d



## USBPHYx\_DEBUG1n field descriptions

Field	Description
31–15 RSVD1	Reserved.
14–13 ENTAILADJVD	Delay increment of the rise of squelch: 00 = Delay is nominal 01 = Delay is +20% 10 = Delay is -20% 11 = Delay is -40%
12–0 RSVD0	Reserved. <b>Note:</b> This bit should remain clear.

### 53.3.10 UTMI RTL Version (USBPHYx\_VERSION)

Fields for RTL Version.

Address: Base address + 80h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	MAJOR								MINOR								STEP															
W																																
Reset	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**USBPHYx\_VERSION field descriptions**

Field	Description
31–24 MAJOR	Fixed read-only value reflecting the MAJOR field of the RTL version.
23–16 MINOR	Fixed read-only value reflecting the MINOR field of the RTL version.
15–0 STEP	Fixed read-only value reflecting the stepping of the RTL version.

## 53.4 USB Analog Memory Map/Register Definition

**USB\_ANALOG memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20C_81A0	USB VBUS Detect Register (USB_ANALOG_USB1_VBUS_DETECT)	32	R/W	0010_0004h	<a href="#">53.4.1/3297</a>
20C_81A4	USB VBUS Detect Register (USB_ANALOG_USB1_VBUS_DETECT_SET)	32	R/W	0010_0004h	<a href="#">53.4.1/3297</a>
20C_81A8	USB VBUS Detect Register (USB_ANALOG_USB1_VBUS_DETECT_CLR)	32	R/W	0010_0004h	<a href="#">53.4.1/3297</a>
20C_81AC	USB VBUS Detect Register (USB_ANALOG_USB1_VBUS_DETECT_TOG)	32	R/W	0010_0004h	<a href="#">53.4.1/3297</a>
20C_81B0	USB Charger Detect Register (USB_ANALOG_USB1_CHRG_DETECT)	32	R/W	0000_0000h	<a href="#">53.4.2/3298</a>
20C_81B4	USB Charger Detect Register (USB_ANALOG_USB1_CHRG_DETECT_SET)	32	R/W	0000_0000h	<a href="#">53.4.2/3298</a>
20C_81B8	USB Charger Detect Register (USB_ANALOG_USB1_CHRG_DETECT_CLR)	32	R/W	0000_0000h	<a href="#">53.4.2/3298</a>

*Table continues on the next page...*

## USB\_ANALOG memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20C_81BC	USB Charger Detect Register (USB_ANALOG_USB1_CHRG_DETECT_TOG)	32	R/W	0000_0000h	<a href="#">53.4.2/3298</a>
20C_81C0	USB VBUS Detect Status Register (USB_ANALOG_USB1_VBUS_DETECT_STAT)	32	R	0000_0000h	<a href="#">53.4.3/3300</a>
20C_81D0	USB Charger Detect Status Register (USB_ANALOG_USB1_CHRG_DETECT_STAT)	32	R	0000_0000h	<a href="#">53.4.4/3302</a>
20C_81F0	USB Misc Register (USB_ANALOG_USB1_MISC)	32	R/W	0000_0002h	<a href="#">53.4.5/3303</a>
20C_81F4	USB Misc Register (USB_ANALOG_USB1_MISC_SET)	32	R/W	0000_0002h	<a href="#">53.4.5/3303</a>
20C_81F8	USB Misc Register (USB_ANALOG_USB1_MISC_CLR)	32	R/W	0000_0002h	<a href="#">53.4.5/3303</a>
20C_81FC	USB Misc Register (USB_ANALOG_USB1_MISC_TOG)	32	R/W	0000_0002h	<a href="#">53.4.5/3303</a>
20C_8200	USB VBUS Detect Register (USB_ANALOG_USB2_VBUS_DETECT)	32	R/W	0010_0004h	<a href="#">53.4.6/3304</a>
20C_8204	USB VBUS Detect Register (USB_ANALOG_USB2_VBUS_DETECT_SET)	32	R/W	0010_0004h	<a href="#">53.4.6/3304</a>
20C_8208	USB VBUS Detect Register (USB_ANALOG_USB2_VBUS_DETECT_CLR)	32	R/W	0010_0004h	<a href="#">53.4.6/3304</a>
20C_820C	USB VBUS Detect Register (USB_ANALOG_USB2_VBUS_DETECT_TOG)	32	R/W	0010_0004h	<a href="#">53.4.6/3304</a>
20C_8210	USB Charger Detect Register (USB_ANALOG_USB2_CHRG_DETECT)	32	R/W	0000_0000h	<a href="#">53.4.7/3306</a>
20C_8214	USB Charger Detect Register (USB_ANALOG_USB2_CHRG_DETECT_SET)	32	R/W	0000_0000h	<a href="#">53.4.7/3306</a>
20C_8218	USB Charger Detect Register (USB_ANALOG_USB2_CHRG_DETECT_CLR)	32	R/W	0000_0000h	<a href="#">53.4.7/3306</a>
20C_821C	USB Charger Detect Register (USB_ANALOG_USB2_CHRG_DETECT_TOG)	32	R/W	0000_0000h	<a href="#">53.4.7/3306</a>
20C_8220	USB VBUS Detect Status Register (USB_ANALOG_USB2_VBUS_DETECT_STAT)	32	R	0000_0000h	<a href="#">53.4.8/3308</a>
20C_8230	USB Charger Detect Status Register (USB_ANALOG_USB2_CHRG_DETECT_STAT)	32	R	0000_0000h	<a href="#">53.4.9/3310</a>
20C_8250	USB Misc Register (USB_ANALOG_USB2_MISC)	32	R/W	0000_0002h	<a href="#">53.4.10/3311</a>
20C_8254	USB Misc Register (USB_ANALOG_USB2_MISC_SET)	32	R/W	0000_0002h	<a href="#">53.4.10/3311</a>
20C_8258	USB Misc Register (USB_ANALOG_USB2_MISC_CLR)	32	R/W	0000_0002h	<a href="#">53.4.10/3311</a>
20C_825C	USB Misc Register (USB_ANALOG_USB2_MISC_TOG)	32	R/W	0000_0002h	<a href="#">53.4.10/3311</a>
20C_8280	Chip Silicon Version (USB_ANALOG_DIGPROG)	32	R	0000_0000h	<a href="#">53.4.11/3312</a>

### 53.4.1 USB VBUS Detect Register (USB\_ANALOG\_USB1\_VBUS\_DETECT<sub>n</sub>)

This register defines controls for USB VBUS detect.

Address: 20C\_8000h base + 1A0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				CHARGE_VBUS	DISCHARGE_VBUS	Reserved				VBUSVALID_PWRUP_CMPS		Reserved			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												VBUSVALID_THRESH			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

#### USB\_ANALOG\_USB1\_VBUS\_DETECT<sub>n</sub> field descriptions

Field	Description
31–28 -	This field is reserved. Reserved.
27 CHARGE_VBUS	USB OTG charge VBUS.
26 DISCHARGE_VBUS	USB OTG discharge VBUS.
25–21 -	This field is reserved. Reserved.
20 VBUSVALID_PWRUP_CMPS	Powers up comparators for vbus_valid detector.
19–3 -	This field is reserved. Reserved.
2–0 VBUSVALID_THRESH	Set the threshold for the VBUSVALID comparator. This comparator is the most accurate method to determine the presence of 5v, and includes hysteresis to minimize the need for software debounce of the detection. This comparator has ~50mV of hysteresis to prevent chattering at the comparator trip point.  000 4V0 — 4.0V

Table continues on the next page...

**USB\_ANALOG\_USB1\_VBUS\_DETECT $n$  field descriptions (continued)**

Field	Description
001	<b>4V1</b> — 4.1V
010	<b>4V2</b> — 4.2V
011	<b>4V3</b> — 4.3V
100	<b>4V4</b> — 4.4V (default)
101	<b>4V5</b> — 4.5V
110	<b>4V6</b> — 4.6V
111	<b>4V7</b> — 4.7V

## 53.4.2 USB Charger Detect Register (USB\_ANALOG\_USB1\_CHRG\_DETECT $n$ )

This register defines controls for USB charger detect.

Address: 20C\_8000h base + 1B0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								Reserved	Reserved		EN_B	CHK_CHRG_B	CHK_CONTACT	Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**USB\_ANALOG\_USB1\_CHRG\_DETECT $n$  field descriptions**

Field	Description
31–24 -	This field is reserved. Reserved.

Table continues on the next page...

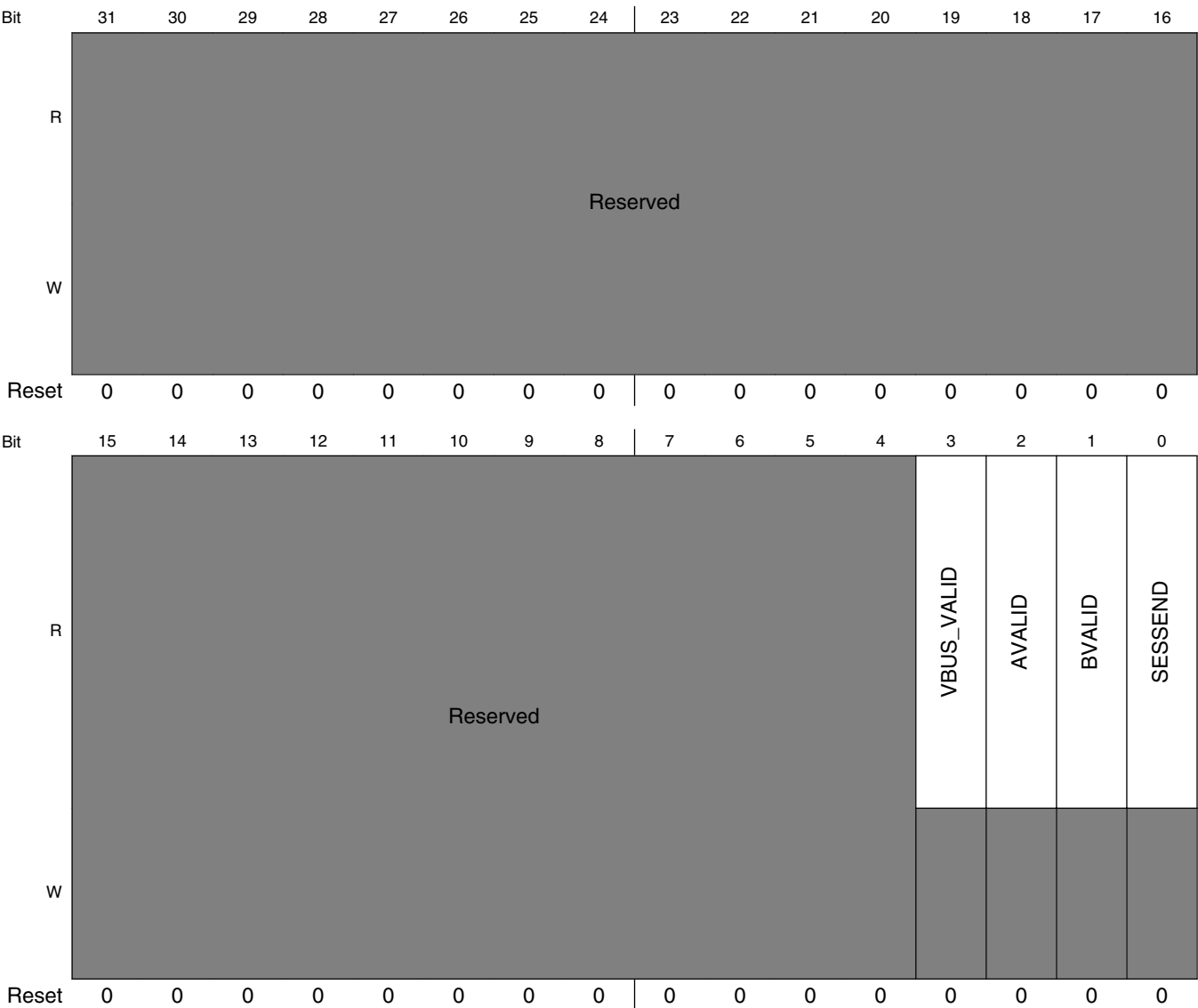
**USB\_ANALOG\_USB1\_CHRG\_DETECT $n$  field descriptions (continued)**

Field	Description
23 -	This field is reserved. Reserved.
22–21 -	This field is reserved. Reserved.
20 EN_B	Control the charger detector.  0 <b>ENABLE</b> — Enable the charger detector. 1 <b>DISABLE</b> — Disable the charger detector.
19 CHK_CHRG_B	0 <b>CHECK</b> — Check whether a charger (either a dedicated charger or a host charger) is connected to USB port. 1 <b>NO_CHECK</b> — Do not check whether a charger is connected to the USB port.
18 CHK_CONTACT	0 <b>NO_CHECK</b> — Do not check the contact of USB plug. 1 <b>CHECK</b> — Check whether the USB plug has been in contact with each other
17–0 -	This field is reserved. Reserved.

### 53.4.3 USB VBUS Detect Status Register (USB\_ANALOG\_USB1\_VBUS\_DETECT\_STAT)

This register defines fields for USB VBUS Detect status.

Address: 20C\_8000h base + 1C0h offset = 20C\_81C0h



USB\_ANALOG\_USB1\_VBUS\_DETECT\_STAT field descriptions

Field	Description
31–4 -	This field is reserved. Reserved.

Table continues on the next page...



**USB\_ANALOG\_USB1\_VBUS\_DETECT\_STAT field descriptions (continued)**

Field	Description
3 VBUS_VALID	VBus valid for USB OTG. This bit is a read only version of the state of the analog signal. It can not be overwritten by software.
2 AVALID	Indicates VBus is valid for a A-peripheral. This bit is a read only version of the state of the analog signal. It can not be overwritten by software.
1 BVALID	Indicates VBus is valid for a B-peripheral. This bit is a read only version of the state of the analog signal. It can not be overwritten by software.
0 SESSEND	Session End for USB OTG. This bit is a read only version of the state of the analog signal. It can not be overwritten by software like the SESSEND bit below.  NOTE: This bit's default value depends on whether VDD5V is present, 0 if VDD5V is present, 1 if VDD5V is not present.

### 53.4.4 USB Charger Detect Status Register (USB\_ANALOG\_USB1\_CHRG\_DETECT\_STAT)

This register defines fields for USB charger detect status.

Address: 20C\_8000h base + 1D0h offset = 20C\_81D0h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												DP_STATE	DM_STATE	CHRG_DETECTED	PLUG_CONTACT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**USB\_ANALOG\_USB1\_CHRG\_DETECT\_STAT field descriptions**

Field	Description
31–4 -	This field is reserved. Reserved.
3 DP_STATE	DP line state output of the charger detector.
2 DM_STATE	DM line state output of the charger detector.
1 CHRG_ DETECTED	State of charger detection. This bit is a read only version of the state of the analog signal. 0 <b>CHARGER_NOT_PRESENT</b> — The USB port is not connected to a charger. 1 <b>CHARGER_PRESENT</b> — A charger (either a dedicated charger or a host charger) is connected to the USB port.
0 PLUG_ CONTACT	State of the USB plug contact detector. 0 <b>NO_CONTACT</b> — The USB plug has not made contact. 1 <b>GOOD_CONTACT</b> — The USB plug has made good contact.

**53.4.5 USB Misc Register (USB\_ANALOG\_USB1\_MISCN)**

This register defines controls for USB.

Address: 20C\_8000h base + 1F0h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved		Reserved													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														EN DEGLITCH	HS_USE_EXTERNAL_R
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

**USB\_ANALOG\_USB1\_MISC<sub>n</sub> field descriptions**

Field	Description
31 -	This field is reserved. Reserved.
30 EN_CLK_UTMI	Enables the clk to the UTMI block.
29–2 -	This field is reserved. Reserved.
1 EN_DEGLITCH	Enable the deglitching circuit of the USB PLL output.
0 HS_USE_EXTERNAL_R	Use external resistor to generate the current bias for the high speed transmitter. This bit should not be changed unless recommended by Freescale.

### 53.4.6 USB VBUS Detect Register (USB\_ANALOG\_USB2\_VBUS\_DETECT<sub>n</sub>)

This register defines controls for USB VBUS detect.

Address: 20C\_8000h base + 200h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved				CHARGE_VBUS	DISCHARGE_VBUS	Reserved					VBUSVALID_PWRUP_CMPS	Reserved			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												VBUSVALID_THRESH			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

**USB\_ANALOG\_USB2\_VBUS\_DETECT<sub>n</sub> field descriptions**

Field	Description
31–28 -	This field is reserved. Reserved.

Table continues on the next page...

**USB\_ANALOG\_USB2\_VBUS\_DETECT<sub>n</sub> field descriptions (continued)**

Field	Description
27 CHARGE_VBUS	USB OTG charge VBUS.
26 DISCHARGE_VBUS	USB OTG discharge VBUS.
25–21 -	This field is reserved. Reserved.
20 VBUSVALID_PWRUP_CMPS	Powers up comparators for vbus_valid detector.
19–3 -	This field is reserved. Reserved.
2–0 VBUSVALID_THRESH	Set the threshold for the VBUSVALID comparator. This comparator is the most accurate method to determine the presence of 5v, and includes hysteresis to minimize the need for software debounce of the detection. This comparator has ~50mV of hysteresis to prevent chattering at the comparator trip point.  000 <b>4V0</b> — 4.0V 001 <b>4V1</b> — 4.1V 010 <b>4V2</b> — 4.2V 011 <b>4V3</b> — 4.3V 100 <b>4V4</b> — 4.4V (default) 101 <b>4V5</b> — 4.5V 110 <b>4V6</b> — 4.6V 111 <b>4V7</b> — 4.7V

### 53.4.7 USB Charger Detect Register (USB\_ANALOG\_USB2\_CHRG\_DETECT $n$ )

This register defines controls for USB charger detect.

Address: 20C\_8000h base + 210h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved								Reserved	Reserved		EN_B	CHK_CHRG_B	CHK_CONTACT	Reserved	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**USB\_ANALOG\_USB2\_CHRG\_DETECT $n$  field descriptions**

Field	Description
31–24 -	This field is reserved. Reserved.
23 -	This field is reserved. Reserved.
22–21 -	This field is reserved. Reserved.
20 EN_B	Control the charger detector.  0 <b>ENABLE</b> — Enable the charger detector. 1 <b>DISABLE</b> — Disable the charger detector.
19 CHK_CHRG_B	0 <b>CHECK</b> — Check whether a charger (either a dedicated charger or a host charger) is connected to USB port. 1 <b>NO_CHECK</b> — Do not check whether a charger is connected to the USB port.

*Table continues on the next page...*

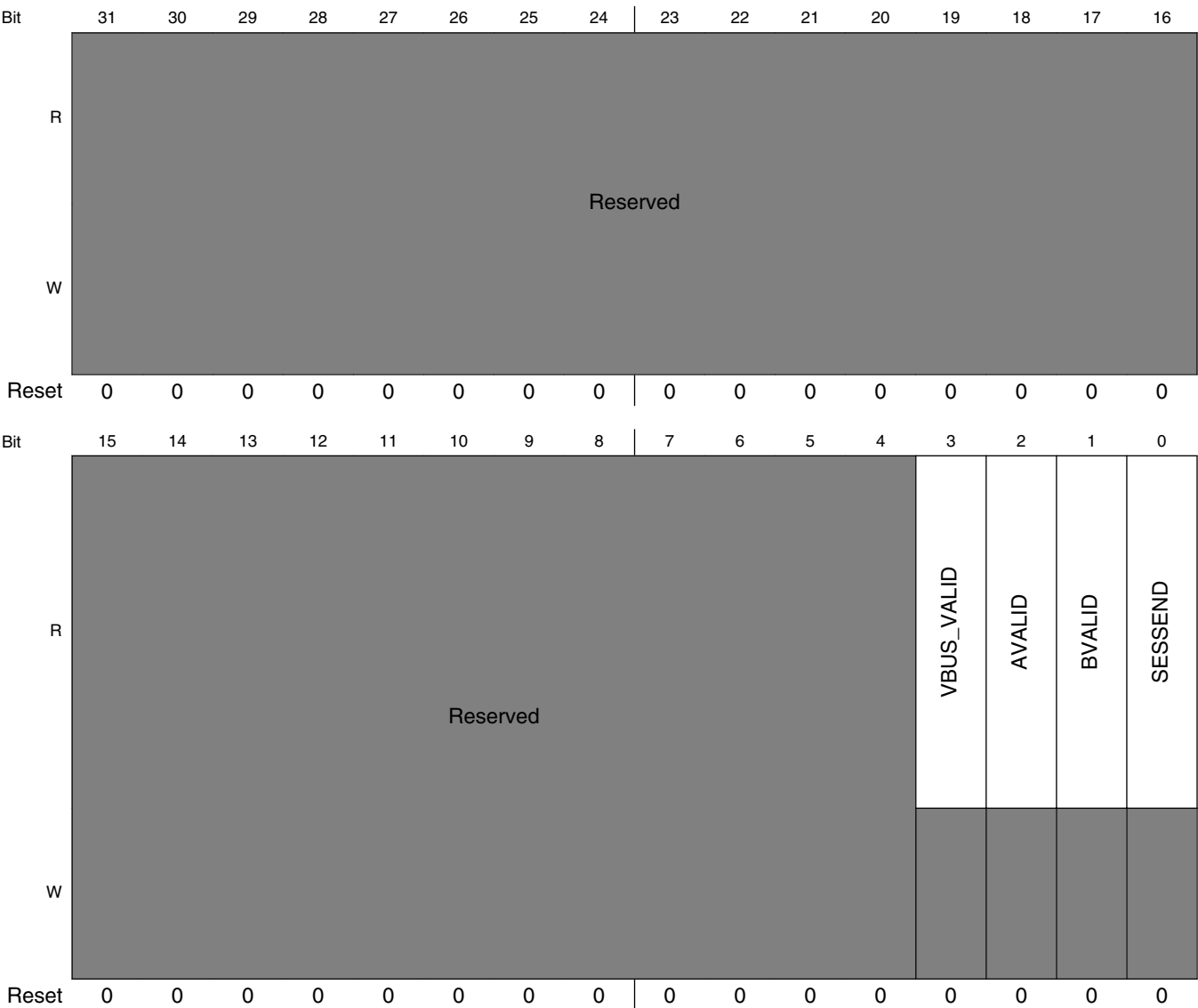
**USB\_ANALOG\_USB2\_CHRG\_DETECT<sub>n</sub> field descriptions (continued)**

Field	Description
18 CHK_CONTACT	0 <b>NO_CHECK</b> — Do not check the contact of USB plug. 1 <b>CHECK</b> — Check whether the USB plug has been in contact with each other
17–0 -	This field is reserved. Reserved.

### 53.4.8 USB VBUS Detect Status Register (USB\_ANALOG\_USB2\_VBUS\_DETECT\_STAT)

This register defines fields for USB VBUS Detect status.

Address: 20C\_8000h base + 220h offset = 20C\_8220h



USB\_ANALOG\_USB2\_VBUS\_DETECT\_STAT field descriptions

Field	Description
31–4 -	This field is reserved. Reserved.

Table continues on the next page...



**USB\_ANALOG\_USB2\_VBUS\_DETECT\_STAT field descriptions (continued)**

Field	Description
3 VBUS_VALID	VBus valid for USB OTG. This bit is a read only version of the state of the analog signal. It can not be overwritten by software.
2 AVALID	Indicates VBus is valid for a A-peripheral. This bit is a read only version of the state of the analog signal. It can not be overwritten by software.
1 BVALID	Indicates VBus is valid for a B-peripheral. This bit is a read only version of the state of the analog signal. It can not be overwritten by software.
0 SESSEND	Session End for USB OTG. This bit is a read only version of the state of the analog signal. It can not be overwritten by software like the SESSEND bit below.  NOTE: This bit's default value depends on whether VDD5V is present, 0 if VDD5V is present, 1 if VDD5V is not present.

### 53.4.9 USB Charger Detect Status Register (USB\_ANALOG\_USB2\_CHRG\_DETECT\_STAT)

This register defines fields for USB charger detect status.

Address: 20C\_8000h base + 230h offset = 20C\_8230h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved												DP_STATE	DM_STATE	CHRG_DETECTED	PLUG_CONTACT
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**USB\_ANALOG\_USB2\_CHRG\_DETECT\_STAT field descriptions**

Field	Description
31–4 -	This field is reserved. Reserved.
3 DP_STATE	DP line state output of the charger detector.
2 DM_STATE	DM line state output of the charger detector.
1 CHRG_ DETECTED	State of charger detection. This bit is a read only version of the state of the analog signal. 0 <b>CHARGER_NOT_PRESENT</b> — The USB port is not connected to a charger. 1 <b>CHARGER_PRESENT</b> — A charger (either a dedicated charger or a host charger) is connected to the USB port.
0 PLUG_ CONTACT	State of the USB plug contact detector. 0 <b>NO_CONTACT</b> — The USB plug has not made contact. 1 <b>GOOD_CONTACT</b> — The USB plug has made good contact.

**53.4.10 USB Misc Register (USB\_ANALOG\_USB2\_MISCn)**

This register defines controls for USB.

Address: 20C\_8000h base + 250h offset + (4d × i), where i=0d to 3d

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	Reserved	EN_CLK_UTMI	Reserved													
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved														EN DEGLITCH	HS_USE_EXTERNAL_R
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

**USB\_ANALOG\_USB2\_MISC<sub>n</sub> field descriptions**

Field	Description
31 -	This field is reserved. Reserved.
30 EN_CLK_UTMI	Enables the clk to the UTMI block.
29–2 -	This field is reserved. Reserved.
1 EN_DEGLITCH	Enable the deglitching circuit of the USB PLL output.
0 HS_USE_EXTERNAL_R	Use external resistor to generate the current bias for the high speed transmitter. This bit should not be changed unless recommended by Freescale.

**53.4.11 Chip Silicon Version (USB\_ANALOG\_DIGPROG)**

The DIGPROG register returns the digital program ID for the silicon.

Address: 20C\_8000h base + 280h offset = 20C\_8280h

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	Reserved								MAJOR								MINOR															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**USB\_ANALOG\_DIGPROG field descriptions**

Field	Description
31–24 -	This field is reserved. Reserved.
23–8 MAJOR	Fixed read-only value reflecting the MAJOR field of the RTL version.
7–0 MINOR	Fixed read-only value reflecting the MINOR field of the RTL version.

## Chapter 54

# Ultra Secured Digital Host Controller (uSDHC)

### 54.1 Overview

The Ultra Secured Digital Host Controller (uSDHC) provides the interface between the host system and the SD/SDIO/MMC cards, as depicted in [Figure 54-1](#).

The uSDHC acts as a bridge, passing host bus transactions to the SD/SDIO/MMC cards by sending commands and performing data accesses to/from the cards.

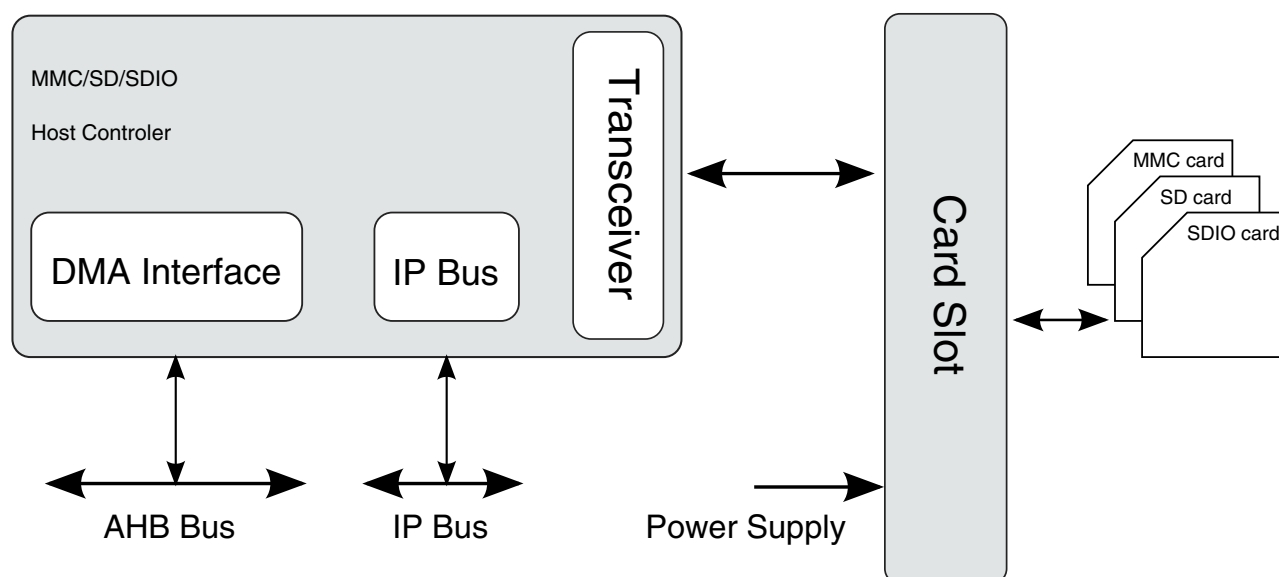
It handles the SD/SDIO/MMC protocols at the transmission level.

The following are brief descriptions of the cards supported by the uSDHC:

The Multi Media Card (MMC) is a universal low cost data storage and communication media designed to cover a wide array of applications including mobile video and gaming. Previous MMC cards were based on a 7-pin serial bus with a single data pin, while the new high speed MMC communication is based on an advanced 11-pin serial bus designed to operate in the low voltage range.

The Secure Digital Card (SD) is an evolution of the old MMC technology. It is specifically designed to meet the security, capacity, performance, and environment requirements inherent in newly-emerging audio and video consumer electronic devices. The physical form factor, pin assignment and data transfer protocol are forward-compatible with the old MMC (with some additions).

Under the SD protocol, it can be categorized into Memory card, I/O card and Combo card, which has both memory and I/O functions. The memory card invokes a copyright protection mechanism that complies with the security of the SDMI standard. The I/O card, which is also known as SDIO card, provides high-speed data I/O with low power consumption for mobile electronic devices. For the sake of simplicity, the following figure does not show cards with reduced size or mini cards.



**Figure 54-1. System Connection of the uSDHC**

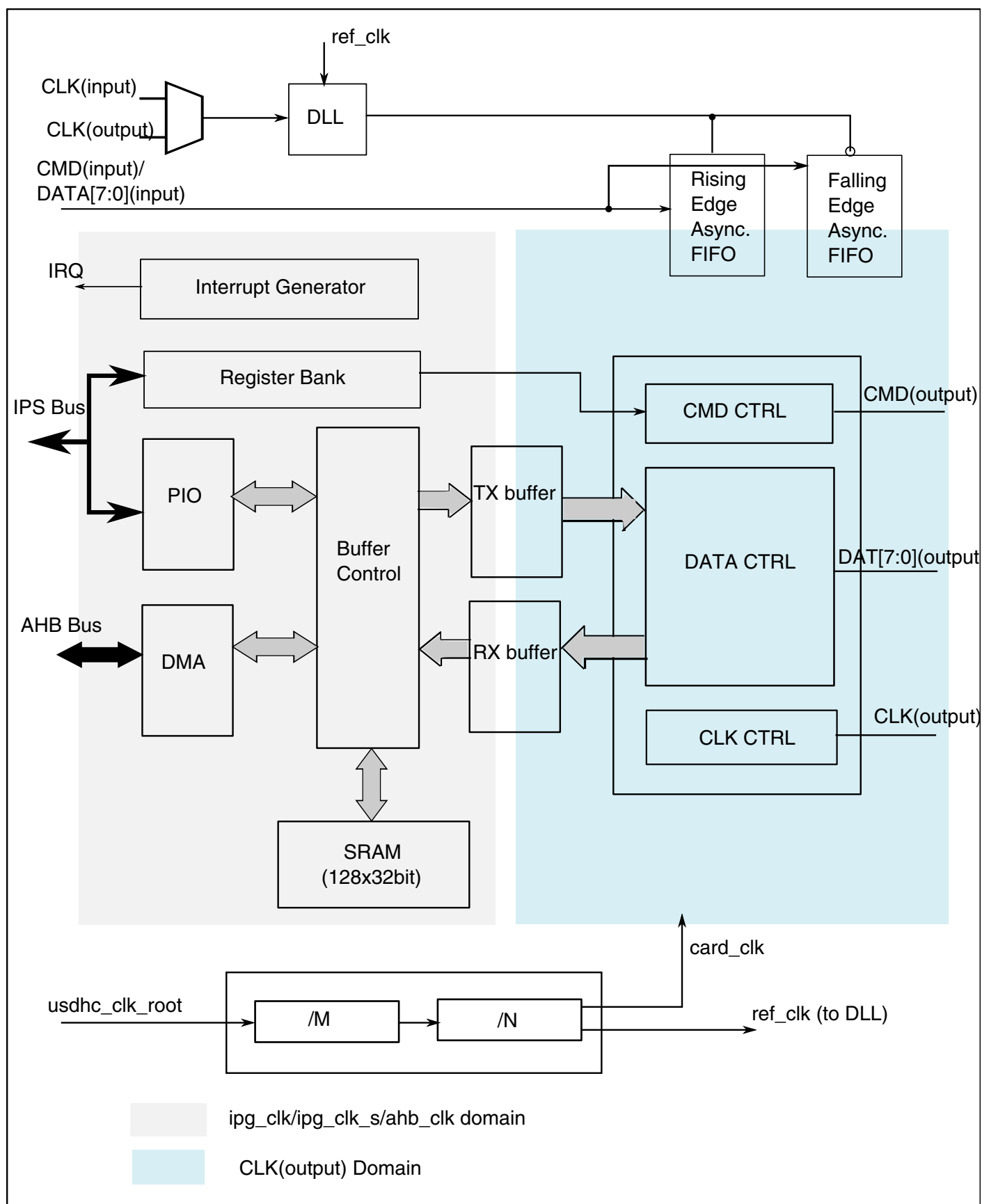


Figure 54-2. ultra Secure Digital Host Controller Block Diagram

## 54.1.1 Features

The features of the uSDHC module include the following:

- Conforms to the SD Host Controller Standard Specification version 3.0
- Compatible with the MMC System Specification version 4.2/4.3/4.4/4.41
- Compatible with the SD Memory Card Specification version 3.0 and supports the Extended Capacity SD Memory Card
- Compatible with the SDIO Card Specification version 3.0
- Designed to work with SD Memory, miniSD Memory, SDIO, miniSDIO, SD Combo, MMC, MMC plus, and MMC RS cards
- Card bus clock frequency up to 208 MHz
- Supports 1-bit / 4-bit SD and SDIO modes, 1-bit / 4-bit / 8-bit MMC modes
  - Up to 832 Mbps of data transfer for SDIO cards using 4 parallel data lines in SDR(Single Data Rate) mode
  - Up to 400 Mbps of data transfer for SDIO card using 4 parallel data lines in DDR(Dual Data Rate) mode
  - Up to 832 Mbps of data transfer for SDXC cards using 4 parallel data lines in SDR(Single Data Rate) mode
  - Up to 400 Mbps of data transfer for SDXC card using 4 parallel data lines in DDR(Dual Data Rate) mode
  - Up to 416 Mbps of data transfer for MMC cards using 8 parallel data lines in SDR(Single Data Rate) mode
  - Up to 832 Mbps of data transfer for MMC cards using 8 parallel data lines in DDR(Dual Data Rate) mode
- Supports single block/multi-block read and write
- Supports block sizes of 1 ~ 4096 bytes
- Supports the write protection switch for write operations
- Supports both synchronous and asynchronous abort
- Supports pause during the data transfer at block gap
- Supports SDIO Read Wait and Suspend Resume operations
- Supports Auto CMD12 for multi-block transfer
- Host can initiate non-data transfer command while data transfer is in progress
- Allows cards to interrupt the host in 1-bit and 4-bit SDIO modes, also supports interrupt period
- Embodies a fully configurable 128x32-bit FIFO for read/write data
- Supports internal and external DMA capabilities
- Support voltage selection by configuring vendor specific register bit
- Supports Advanced DMA to perform linked memory access



## 54.1.2 Modes and Operations

### 54.1.2.1 Data transfer Modes

The uSDHC can select the following modes for data transfer:

- SD 1-bit
- SD 4-bit
- MMC 1-bit
- MMC 4-bit
- MMC 8-bit
- Identification Mode (up to 400 kHz)
- MMC full speed mode (up to 20 MHz)
- MMC high speed mode (up to 52 MHz)
- MMC HS200 mode(up to 200Mhz)
- MMC DDR mode (52MHz both edges)
- SD/SDIO full speed mode (up to 25 MHz)
- SD/SDIO high speed mode (up to 50 MHz)
- SD/SDIO UHS-I mode(up to 208Mhz in SDR mode, up to 50Mhz in DDR mode)

## 54.2 External Signals

The following table describes the external signals of USDHC:

**Table 54-1. USDHC1 External Signals**

Signal	Description	Pad	Mode	Direction
SD1_CD_B (CD_B)	Card detection pin If not used(for the embedded memory),tie low to indicate there is a card attached.	ECSPi2_SS0	ALT4	I
		FEC_TXD1	ALT3	
		KEY_COL0	ALT4	
		KEY_ROW7	ALT6	
SD1_CLK (CLK)	Clock for MMC/SD/SDIO card	SD1_CLK	ALT0	O
SD1_CMD (CMD)	CMD line connect to card	SD1_CMD	ALT0	IO
SD1_DATA0 (DATA0)	DATA0 line in all modes Also used to detect busy state	SD1_DAT0	ALT0	IO
SD1_DATA1 (DATA1)	DATA1 line in 4/8-bit mode Also used to detect interrupt in 1/4-bit mode	SD1_DAT1	ALT0	IO

*Table continues on the next page...*

**Table 54-1. USDHC1 External Signals (continued)**

Signal	Description	Pad	Mode	Direction
SD1_DATA2 (DATA2)	DATA2 line or Read Wait in 4-bit mode Read Wait in 1-bit mode	SD1_DAT2	ALT0	IO
SD1_DATA3 (DATA3)	DATA3 line in 4/8-bit mode or configured as card detection pin May be configured as card detection pin in 1-bit mode	SD1_DAT3	ALT0	IO
SD1_DATA4 (DATA4)	DATA4 line in 8-bit mode, not used in other modes	SD1_DAT4	ALT0	IO
SD1_DATA5 (DATA5)	DATA5 line in 8-bit mode, not used in other modes	SD1_DAT5	ALT0	IO
SD1_DATA6 (DATA6)	DATA6 line in 8-bit mode, not used in other modes	SD1_DAT6	ALT0	IO
SD1_DATA7 (DATA7)	DATA7 line in 8-bit mode, not used in other modes	SD1_DAT7	ALT0	IO
SD1_LCTL (LCTL)	LED control used to drive an external LED Active high Fully controlled by the driver Optional output	AUD_RXD	ALT4	O
		REF_CLK_32K	ALT4	
SD1_RESET (RESET)	Card hardware reset signal, active LOW	ECSPI2_SCLK	ALT4	O
		FEC_MDC	ALT3	
		KEY_COL3	ALT6	
SD1_VSELECT (VSELECT)	IO power voltage selection signal	ECSPI2_MOSI	ALT4	O
		FEC_RXD0	ALT3	
		KEY_ROW3	ALT6	
		SD3_DAT1	ALT4	
SD1_WP (WP)	Card write protect detect If not used(for the embedded memory), tie low to indicate it's not write protected.	ECSPI2_MISO	ALT4	I
		FEC_TX_EN	ALT3	
		KEY_COL7	ALT6	
		KEY_ROW0	ALT4	

**Table 54-2. USDHC2 External Signals**

Signal	Description	Pad	Mode	Direction
SD2_CD_B (CD_B)	Card detection pin If not used(for the embedded memory),tie low to indicate there is a card attached.	ECSPI1_SS0	ALT4	I
		EPDC_GDSP	ALT6	
		SD2_DAT7	ALT4	
SD2_CLK (CLK)	Clock for MMC/SD/SDIO card	SD2_CLK	ALT0	O
SD2_CMD (CMD)	CMD line connect to card	SD2_CMD	ALT0	IO
SD2_DATA0 (DATA0)	DATA0 line in all modes Also used to detect busy state	SD2_DAT0	ALT0	IO

Table continues on the next page...

**Table 54-2. USDHC2 External Signals (continued)**

Signal	Description	Pad	Mode	Direction
SD2_DATA1 (DATA1)	DATA1 line in 4/8-bit mode Also used to detect interrupt in 1/4-bit mode	SD2_DAT1	ALT0	IO
SD2_DATA2 (DATA2)	DATA2 line or Read Wait in 4-bit mode Read Wait in 1-bit mode	SD2_DAT2	ALT0	IO
SD2_DATA3 (DATA3)	DATA3 line in 4/8-bit mode or configured as card detection pin May be configured as card detection pin in 1-bit mode	SD2_DAT3	ALT0	IO
SD2_DATA4 (DATA4)	DATA4 line in 8-bit mode, not used in other modes	SD2_DAT4	ALT0	IO
SD2_DATA5 (DATA5)	DATA5 line in 8-bit mode, not used in other modes	SD2_DAT5	ALT0	IO
SD2_DATA6 (DATA6)	DATA6 line in 8-bit mode, not used in other modes	SD2_DAT6	ALT0	IO
SD2_DATA7 (DATA7)	DATA7 line in 8-bit mode, not used in other modes	SD2_DAT7	ALT0	IO
SD2_LCTL (LCTL)	LED control used to drive an external LED Active high Fully controlled by the driver Optional output	AUD_TXC	ALT4	O
SD2_RESET (RESET)	Card hardware reset signal, active LOW	ECSPI1_SCLK	ALT4	O
		EPDC_GDCLK	ALT6	
		SD2_RST	ALT0	
SD2_VSELECT (VSELECT)	IO power voltage selection signal	ECSPI1_MOSI	ALT4	O
		EPDC_GDOE	ALT6	
SD2_WP (WP)	Card write protect detect If not used(for the embedded memory), tie low to indicate it's not write protected.	ECSPI1_MISO	ALT4	I
		EPDC_GDRL	ALT6	
		SD2_DAT6	ALT4	

**Table 54-3. USDHC3 External Signals**

Signal	Description	Pad	Mode	Direction
SD3_CD_B (CD_B)	Card detection pin If not used(for the embedded memory),tie low to indicate there is a card attached.	EPDC_PWRWAKEUP	ALT6	I
		FEC_TXD1	ALT4	
		I2C2_SDA	ALT4	
		REF_CLK_32K	ALT6	
SD3_CLK (CLK)	Clock for MMC/SD/SDIO card	SD3_CLK	ALT0	O
SD3_CMD (CMD)	CMD line connect to card	SD3_CMD	ALT0	IO

Table continues on the next page...

**Table 54-3. USDHC3 External Signals (continued)**

Signal	Description	Pad	Mode	Direction
SD3_DATA0 (DATA0)	DATA0 line in all modes Also used to detect busy state	SD3_DAT0	ALT0	IO
SD3_DATA1 (DATA1)	DATA1 line in 4/8-bit mode Also used to detect interrupt in 1/4-bit mode	SD3_DAT1	ALT0	IO
SD3_DATA2 (DATA2)	DATA2 line or Read Wait in 4-bit mode Read Wait in 1-bit mode	SD3_DAT2	ALT0	IO
SD3_DATA3 (DATA3)	DATA3 line in 4/8-bit mode or configured as card detection pin May be configured as card detection pin in 1-bit mode	SD3_DAT3	ALT0	IO
SD3_DATA4 (DATA4)	DATA4 line in 8-bit mode, not used in other modes	KEY_COL1	ALT4	IO
		SD2_DAT4	ALT1	
SD3_DATA5 (DATA5)	DATA5 line in 8-bit mode, not used in other modes	KEY_ROW1	ALT4	IO
		SD2_DAT5	ALT1	
SD3_DATA6 (DATA6)	DATA6 line in 8-bit mode, not used in other modes	KEY_COL2	ALT4	IO
		SD2_DAT6	ALT1	
SD3_DATA7 (DATA7)	DATA7 line in 8-bit mode, not used in other modes	KEY_ROW2	ALT4	IO
		SD2_DAT7	ALT1	
SD3_LCTL (LCTL)	LED control used to drive an external LED Active high Fully controlled by the driver Optional output	AUD_TXFS	ALT4	O
SD3_RESET (RESET)	Card hardware reset signal, active LOW	EPDC_PWRCOM	ALT6	O
		FEC_MDC	ALT4	
		I2C1_SCL	ALT4	
		KEY_COL6	ALT6	
SD3_VSELECT (VSELECT)	IO power voltage selection signal	EPDC_PWRINT	ALT6	O
		FEC_RXD0	ALT4	
		I2C1_SDA	ALT4	
		KEY_ROW6	ALT6	
SD3_WP (WP)	Card write protect detect If not used(for the embedded memory), tie low to indicate it's not write protected.	EPDC_PWRSTAT	ALT6	I
		FEC_TX_EN	ALT4	
		I2C2_SCL	ALT4	
		REF_CLK_24M	ALT6	

**Table 54-4. USDHC4 External Signals**

Signal	Description	Pad	Mode	Direction
SD4_CLK (CLK)	Clock for MMC/SD/SDIO card	SD4_CLK	ALT0	O

Table continues on the next page...

**Table 54-4. USDHC4 External Signals (continued)**

Signal	Description	Pad	Mode	Direction
SD4_CMD (CMD)	CMD line connect to card	SD4_CMD	ALT0	IO
SD4_DATA0 (DATA0)	DATA0 line in all modes Also used to detect busy state	SD4_DAT0	ALT1	IO
SD4_DATA1 (DATA1)	DATA1 line in 4/8-bit mode Also used to detect interrupt in 1/4-bit mode	SD4_DAT1	ALT1	IO
SD4_DATA2 (DATA2)	DATA2 line or Read Wait in 4-bit mode Read Wait in 1-bit mode	SD4_DAT2	ALT1	IO
SD4_DATA3 (DATA3)	DATA3 line in 4/8-bit mode or configured as card detection pin May be configured as card detection pin in 1-bit mode	SD4_DAT3	ALT1	IO
SD4_DATA4 (DATA4)	DATA4 line in 8-bit mode, not used in other modes	SD4_DAT4	ALT1	IO
SD4_DATA5 (DATA5)	DATA5 line in 8-bit mode, not used in other modes	SD4_DAT5	ALT1	IO
SD4_DATA6 (DATA6)	DATA6 line in 8-bit mode, not used in other modes	SD4_DAT6	ALT1	IO
SD4_DATA7 (DATA7)	DATA7 line in 8-bit mode, not used in other modes	SD4_DAT7	ALT1	IO
SD4_RESET (RESET)	Card hardware reset signal, active LOW	NANDF_ALE	ALT1	O
SD4_VSELECT (VSELECT)	IO power voltage selection signal	NANDF_CS1	ALT1	O

**Table 54-5. USDHC4 External Signals**

Signal	Description	Pad	Mode	Direction
SD4_CD_B (CD_B)	Card detection pin If not used(for the embedded memory),tie low to indicate there is a card attached.	EPDC_D11	ALT6	I
		EPDC_PWRCTRL3	ALT6	
SD4_CLK (CLK)	Clock for MMC/SD/SDIO card	EPDC_BDR0	ALT1	O
		FEC_MDIO	ALT1	
		KEY_COL4	ALT4	
SD4_CMD (CMD)	CMD line connect to card	EPDC_BDR1	ALT1	IO
		FEC_TX_CLK	ALT1	
		KEY_ROW4	ALT4	
SD4_DATA0 (DATA0)	DATA0 line in all modes Also used to detect busy state	EPDC_PWRCOM	ALT1	IO
		FEC_RX_ER	ALT1	
		KEY_COL5	ALT4	

Table continues on the next page...

**Table 54-5. USDHC4 External Signals (continued)**

Signal	Description	Pad	Mode	Direction
SD4_DATA1 (DATA1)	DATA1 line in 4/8-bit mode Also used to detect interrupt in 1/4-bit mode	EPDC_PWRINT	ALT1	IO
		FEC_CRS_DV	ALT1	
		KEY_ROW5	ALT4	
SD4_DATA2 (DATA2)	DATA2 line or Read Wait in 4-bit mode	EPDC_PWRSTAT	ALT1	IO
		FEC_RXD1	ALT1	
	Read Wait in 1-bit mode	KEY_COL6	ALT4	
SD4_DATA3 (DATA3)	DATA3 line in 4/8-bit mode or configured as card detection pin	EPDC_PWRWAKEUP	ALT1	IO
		FEC_TXD0	ALT1	
	May be configured as card detection pin in 1-bit mode	KEY_ROW6	ALT4	
SD4_DATA4 (DATA4)	DATA4 line in 8-bit mode, not used in other modes	FEC_MDC	ALT1	IO
		KEY_COL7	ALT4	
		LCD_CLK	ALT1	
SD4_DATA5 (DATA5)	DATA5 line in 8-bit mode, not used in other modes	FEC_RXD0	ALT1	IO
		KEY_ROW7	ALT4	
		LCD_ENABLE	ALT1	
SD4_DATA6 (DATA6)	DATA6 line in 8-bit mode, not used in other modes	FEC_TX_EN	ALT1	IO
		KEY_COL3	ALT4	
		LCD_HSYNC	ALT1	
SD4_DATA7 (DATA7)	DATA7 line in 8-bit mode, not used in other modes	FEC_TXD1	ALT1	IO
		KEY_ROW3	ALT4	
		LCD_VSYNC	ALT1	
SD4_LCTL (LCTL)	LED control used to drive an external LED Active high Fully controlled by the driver Optional output	AUD_TXD	ALT4	O
SD4_RESET (RESET)	Card hardware reset signal, active LOW	EPDC_D8	ALT6	O
		EPDC_PWRCTRL0	ALT6	
		FEC_REF_CLK	ALT1	
SD4_VSELECT (VSELECT)	IO power voltage selection signal	EPDC_D9	ALT6	O
		EPDC_PWRCTRL1	ALT6	
SD4_WP (WP)	Card write protect detect If not used(for the embedded memory), tie low to indicate it's not write protected.	EPDC_D10	ALT6	I
		EPDC_PWRCTRL2	ALT6	

## 54.2.1 Signals Overview

The uSDHC has 14 associated I/O signals.

- The CLK is an internally generated clock used to drive the MMC, SD, SDIO cards.
- The CMD I/O is used to send commands and receive responses to and from the card. Eight data lines (DAT7~DAT0) are used to perform data transfers between the uSDHC and the card.
- The CD and WP are card detection and write protection signals directly routed from the socket. These two signals are active low (0). A low on CD# means that a card is inserted, and a high on WP means that the write protect switch is active.
- LCTL is an output signal used to drive an external LED to indicate that the SD interface is busy.
- RST is an output signal used to reset the MMC card.
- VSELECT is an output signal used to change the voltage of the external power supplier.

CD, WP, LCTL, RST and VSELECT are all optional for system implementation. If the uSDHC needs to support a 4-bit data transfer, DAT7~DAT4 can also be optional and tied to high.

### 54.3 Clocks

The table found here describes the clock sources for uSDHC.

Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 54-6. uSDHC Clocks**

Clock name	Clock Root	Description
hclk	ahb_clk_root	AHB bus clock
ipg_clk	ipg_clk_root	Peripheral clock
ipg_clk_perclk	usdhc_clk_root	Base clock
ipg_clk_s	ipg_clk_root	Peripheral access clock for register accesses

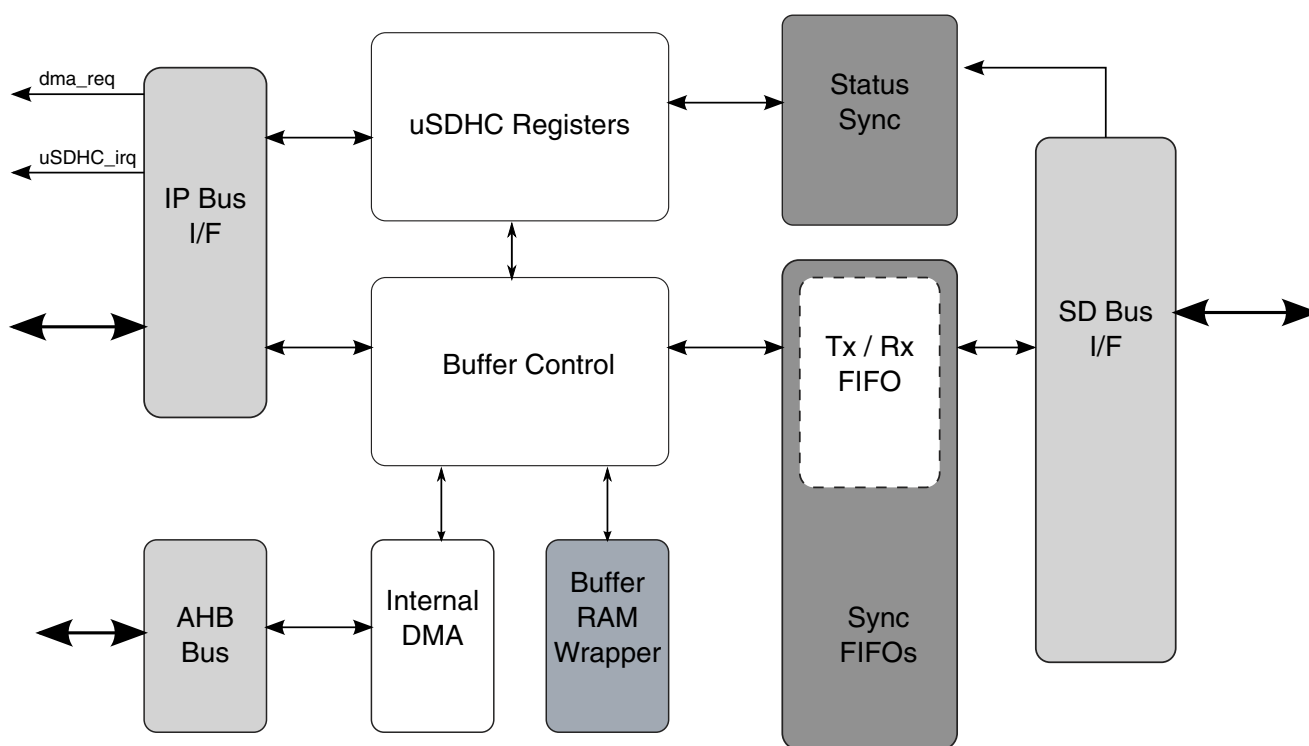
### 54.4 Functional Description

The following sections provide a brief functional description of the major system blocks, including the Data Buffer, DMA AHB interface, register bank as well as IP Bus interface, dual-port memory wrapper, data/command controller, clock & reset manager and clock generator.

### 54.4.1 Data Buffer

The uSDHC uses one configurable data buffer to transfer data between the system bus (IP Bus or AHB Bus) and the SD card in an optimized manner, maximizing throughput between the two clock domains (IP peripheral clock and the master clock).

The buffer is used as temporary storage for data being transferred between the host system and the card. The watermark levels for read and write are both configurable and can be from 1 to 128 words. The burst lengths for read and write are also configurable and can be from 1 to 31 words.



### Figure 54-3. uSDHC Buffer Scheme

There are 3 transfer modes to access the data buffer:

- CPU polling mode:
  - For a host read operation, when the number of words received in the buffer meets or exceeds the RD\_WML watermark value, by polling the BRR bit, the Host Driver can read the Buffer Data Port register to fetch the amount of words set in the RD\_WML register from the buffer. The write operation is similar.
- External DMA mode:
  - For a read operation, when there are more words received in the buffer than the amount set in the RD\_WML register, a DMA request is sent out to inform the



external DMA to fetch the data. The request will be immediately de-asserted when there is an access on the Buffer Data Port register. If the number of words in the buffer after the current burst meets or exceeds RD\_WML value, the DMA request is asserted again. For instance, if there are twice as many words in the buffer as there are in the RD\_WML value, there are two successive DMA requests with only one cycle of de-assertion between. The write operation is similar. Note the accesses CPU polling mode and external DMA mode both use the IP bus, and if the external DMA is enabled, in both modes an external DMA request is sent when the buffer is ready.

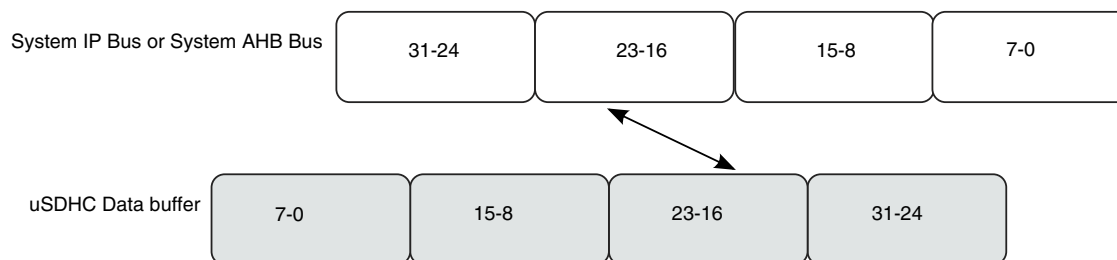
- Internal DMA mode (includes simple and advanced DMA accesses):
  - The internal DMA access, either by simple or advanced DMA, is over the AHB bus. For internal DMA access mode, the external DMA request will never be sent out.

For a read operation, when there are more words in the buffer than the amount set in the RD\_WML register, the internal DMA starts fetching data over the AHB bus. Except for INCR4 and INCR8, the burst type is always INCR mode and the burst length depends on the shortest of following factors:

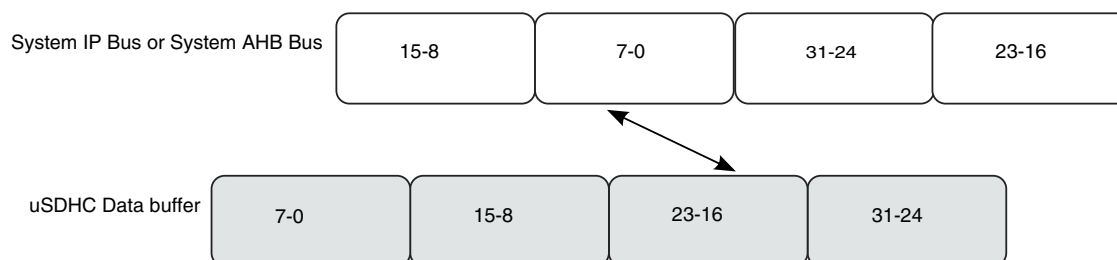
- Burst length configured in the burst length field of the Watermark Level register
- Watermark Level boundary
- Block size boundary
- Data boundary configured in the current descriptor (if the ADMA is active)
- 1 Kbyte address boundary defined in the AHB protocol

Write operation is similar.

Sequential and contiguous access is necessary to ensure the pointer address value is correct. Random or skipped access is not possible. The byte order, by reset, is little endian mode. The actual byte order is swapped inside the buffer, according to the endian mode configured by software (see the following figures). For a host write operation, byte order is swapped after data is fetched from the buffer and ready to send to the SD Bus. For a host read operation, byte order is swapped before the data is stored in the buffer.



**Figure 54-4. Data Swap between System Bus and uSDHC Data Buffer in Byte Little Endian Mode**



**Figure 54-5. Data Swap between System Bus and uSDHC Data Buffer in Half Word Big Endian Mode**

### 54.4.1.1 Write Operation Sequence

There are three ways to write data into the buffer when the user transfers data to the card:

- External DMA through the uSDHC DMA request signal
- Processor core polling through the BWR bit in Interrupt Status register (interrupt or polling)
- Internal DMA

When the internal DMA is not used, (the DMAEN bit in the Transfer Type register is not set when the command is sent), the uSDHC asserts a DMA request when the amount of buffer space exceeds the value set in the WR\_WML register, and is ready for receiving new data. At the same time, the uSDHC sets the BWR bit. The buffer write ready interrupt will be generated if it is enabled by software.

When internal DMA is used, the uSDHC will not inform the system before all the required number of bytes are transferred (if no error was encountered). When an error occurs during the data transfer, the uSDHC will abort the data transfer and abandon the current block. The Host Driver should read the contents of the DMA System Address register to obtain the starting address of the abandoned data block. If the current data

transfer is in multi-block mode, the uSDHC will not automatically send CMD12, even though the AC12EN bit in the Transfer Type register is set. The Host Driver sends CMD12 in this scenario and re-starts the write operation from that address. It is recommended that a Software Reset for Data be applied before the transfer is re-started.

The uSDHC will not start data transmission until the number of words set in the WR\_WML register can be held in the buffer. If the buffer is empty and the Host System does not write data in time, the uSDHC will stop the CLK to avoid the data buffer under-run situation.

#### 54.4.1.2 Read Operation Sequence

There are three ways to read data from the buffer when the user transfers data to the card:

- External DMA through the uSDHC DMA request signal
- Processor core polling through the BRR bit in Interrupt Status register (interrupt or polling)
- Internal DMA

When internal DMA is not used (DMAEN bit in Transfer Type register is not set when the command is sent), the uSDHC asserts a DMA request when the amount of data exceeds the value set in the RD\_WML register, that is available and ready for system fetching data. At the same time, the uSDHC sets the BRR bit. The buffer read ready interrupt will be generated if it is enabled by software.

When internal DMA is used, the uSDHC will not inform the system before all the required number of bytes are transferred (if no error was encountered). When an error occurs during the data transfer, the uSDHC will abort the data transfer and abandon the current block. The Host Driver should read the content of the DMA System Address register to get the starting address of the abandoned data block. If the current data transfer is in multi-block mode, the uSDHC will not automatically send CMD12, even though the AC12EN bit in the Transfer Type register is set. The Host Driver sends CMD12 in this scenario and re-starts the read operation from that address. It is recommended that a Software Reset for Data be applied before the transfer is re-started.

For any write transfer mode, the uSDHC will not start data transmission until the number of words set in the RD\_WML register are in the buffer. If the buffer is full and the Host System does not read data in time, the uSDHC will stop the CLK to avoid the data buffer over-run situation.

### 54.4.1.3 Data Buffer and Block Size

The user needs to know the buffer size for the buffer operation during a data transfer to utilize it in the most optimized way. In the uSDHC, the only data buffer can hold up to 128 words (32-bit) and the watermark levels for write and read can be configured accordingly.

For both read and write, the watermark level can be from 1 to 128 words. For both read and write the burst length can be from 1 to 31 words. The Host Driver may configure the value according to the system situation and requirement.

During a multi-block data transfer, the block length can be set to any value between 1 and 4096 bytes, satisfying the requirements of the external card. The only restriction is from the external card, which can be limited in size or support of a partial block access (which is not the integer times of 512 bytes).

As uSDHC treats each block individually, for block sizes which are not multiples of four (not word-aligned) stuffed bytes are required at the end of each block. For example, if the block size is 7 bytes and there are 12 blocks to write, the system side must write two times for each block. For each block the ending byte will be abandoned by uSDHC because it only sends 7 bytes to the card and picks data from the following system write, resulting in 24 beats of write access in total.

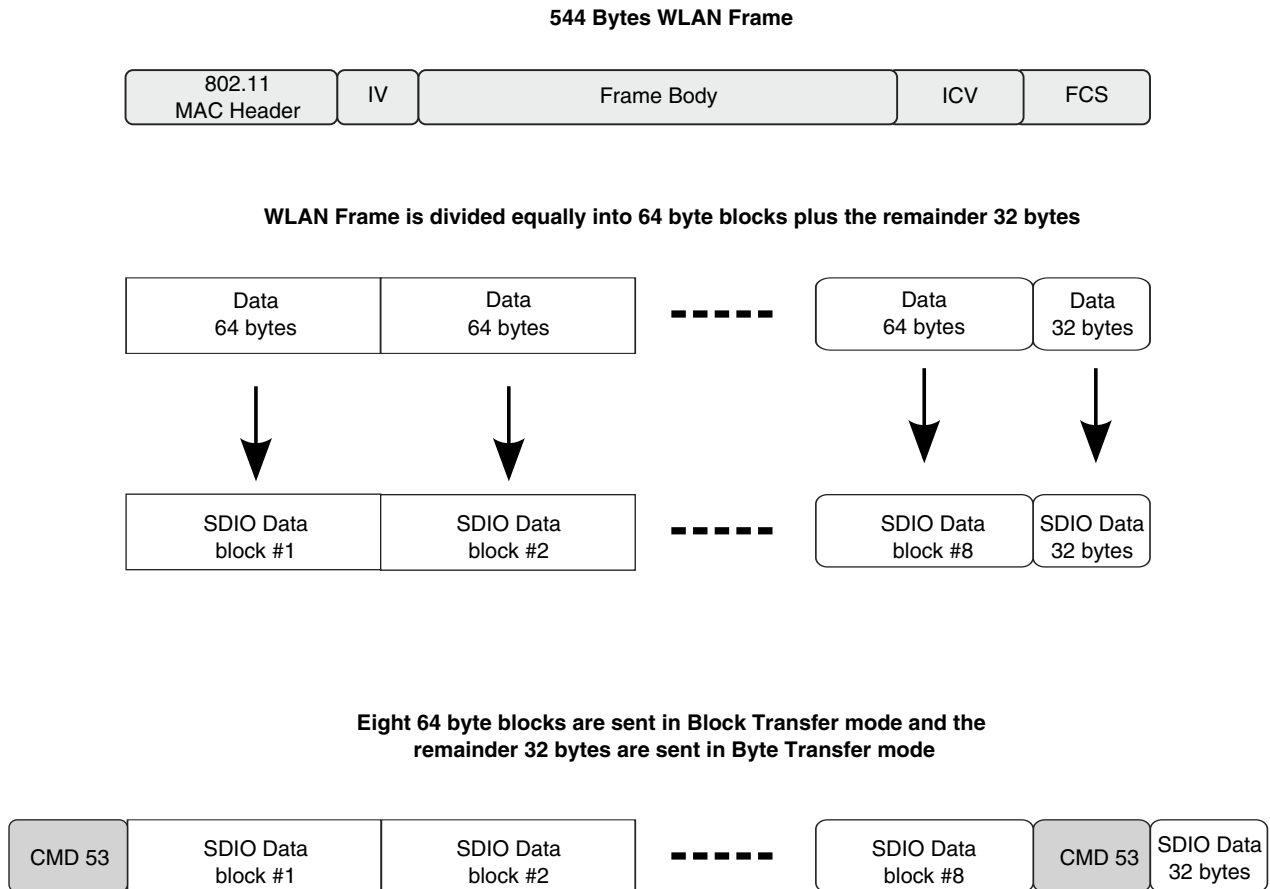
### 54.4.1.4 Dividing Large Data Transfer

This SDIO command CMD53 definition limits the maximum data size of data transfers according to the following formula:

Max data size = Block size x Block count

The length of a multiple block transfer needs to be in block size units. If the total data length can't be divided evenly into a multiple of the block size, then there are two ways to transfer the data which depend on the function and the card design. Option 1 is for the Host Driver to split the transaction. The remainder of the block size data is then transferred by using a single block command at the end. Option 2 is to add dummy data in the last block to fill the block size. For option 2, the card must manage the removal of the dummy data.

See the figure below for an example showing the dividing of large data transfers, assuming a kind of WLAN SDIO card that only supports a block size up to 64 bytes. Although the uSDHC supports a block size of up to 4096 bytes, the SDIO can only accept a block size less than 64 bytes, so the data must be divided (see example below).



**Figure 54-6. Example for Dividing Large Data Transfers**

#### 54.4.1.5 External DMA Request

When the internal DMA is not in use and external DMA is enabled, the Data Buffer will generate a DMA request to the system. During a write operation, when the number of WR\_WML words can be held in the buffer free space, the signal uSDHC\_dreq\_b is asserted to 0, informing the Host System of a DMA write.

The BWR bit in the Interrupt Status register is also set, as long as the BWRSEN bit in the Interrupt Status Enable register is set. The DMA request is de-asserted after several accesses to the Data Port register are made while the buffer's free space can't meet the watermark condition (free space > write watermark level).

On read operation, when the number of RD\_WML words are already in the buffer, the signal uSDHC\_dreq\_b is asserted to 0, informing the Host System for a DMA read. The BRR bit in the Interrupt Status register is also set, as long as the BRRSEN bit in the

Interrupt Status Enable register is set. The DMA request is de-asserted after several accesses to the Data Port register are made while the buffer's data can't meet the watermark condition (the number of data in buffer > read watermark level).

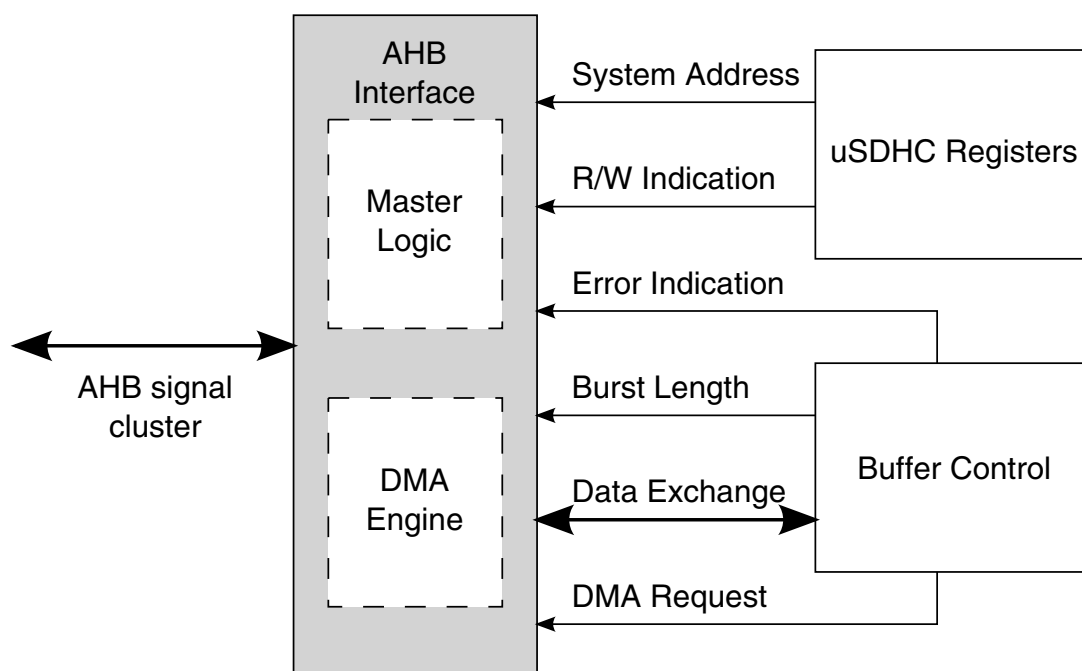
If the DMA burst length can't change during a data transfer for an external DMA transfer, the watermark level (read or write) must be a divisor of the block size. If it is not, transferring the block may cause buffer under-run (read operation) or over-run (write operation). For example, if the block size is 512 bytes, the watermark level of read (or write) must be a power of two between 1 and 128. For processor core polling access there is no such issue, as the last access in the block transfer can be controlled by software. The watermark level can be any value, even larger than the block size (but no greater than 128 words) because the actual number of bytes transferred by the software can be controlled and does not exceed the block size in each transfer.

The uSDHC also supports non-word aligned block size, as long as the card supports that block size. In this case, the watermark level should be set as the number of words. For example, if the block size is 31 bytes, the watermark level can be set to any number of words. For this case, the BLKSIZE bits of the Block Attribute register will be set as 1fh. For the CPU polling access, the burst length can be 1 to 128 words, without restriction. This is because the software will transfer 8 words, and the uSDHC will also set the BWR or BRR bits when the remaining data does not violate data buffer. See [DMA Burst Length](#) for more details about the dynamic watermark level of the data buffer. For the above example, even though 8 words are transferred via the Data Port register, the uSDHC will transfer only 31 bytes over the SD Bus, as required by the BLKSIZE bits. In this data transfer, with non-word aligned block size, the endian mode should be set cautiously or invalid data will be transferred to and from the card.

## 54.4.2 DMA AHB Interface

The internal DMA implements a DMA engine and the AHB master. When the internal DMA is enabled, the uSDHC\_dreq\_b will not be asserted during the transfer, but the BWR and BRR bits will be set if the BWRSEN and BRRSEN bits have been set in the Interrupt Status Enable register.

See the figure below for an illustration of the DMA AHB interface block.



**Figure 54-7. DMA AHB Interface Block**

### 54.4.2.1 Internal DMA Request

If the watermark level requirement is met in data transfer or if the last data of current block is ready in the data buffer, and the Internal DMA is enabled, the Data Buffer block will send a DMA request to AHB interface. Meanwhile, the external DMA request signal (uSDHC\_dreq\_b) is disabled.

The delay in response from the internal DMA engine depends on the system AHB bus loading and the priority assigned to the uSDHC. The DMA engine does not respond to the request during its burst transfer, but is ready to serve as soon as the burst is over. The Data Buffer de-asserts the request if the data buffer space(for write) or bytes in data buffer is smaller than the watermark level. Upon access to the buffer by internal DMA,

the Data Buffer updates its internal buffer pointer, and when the watermark level is satisfied or the last data of current block is ready in the data buffer, another DMA request is sent.

The data transfer is in the block unit, and the subsequent watermark level is always set as the remaining number of words. For instance, for a multi block data read with each block size of 31 bytes, and the burst length set to 6 words. After the first burst transfer, if there are more than 2 words in the buffer (which might contain some data of the next block), another DMA request is sent. This is because the remaining number of words to send for the current block is  $(31 - 6 * 4) / 4 = 2$ . The uSDHC will read 2 words out of the buffer, with 7 valid bytes and 1 stuffed byte.

### **54.4.2.2 DMA Burst Length**

Just like a CPU polling access, the DMA burst length for the internal DMA engine can be from 1 to 16 words. The actual burst length for the DMA depends on the lesser of the configured burst length or the remaining words of the current block.

See the example in [Internal DMA Request](#). After 6 words are read, the burst length will be 2 words, then the next burst length will be 6 words. This is because the next block starts, which is 31 bytes, more than 6 words. The Host Driver may take this variable burst length into account. It is also acceptable to configure the burst length as the divisor of the block size, so that each time the burst length will be the same.

### **54.4.2.3 AHB Master Interface**

It is possible that the internal AHB DMA engine could fail during the data transfer. Upon detection of an AHB bus error during DMA transfer, the DMA engine stops the transfer and goes to the idle state. At that point, the internal data buffer stops receiving incoming data and sending out data. The DMAE bit in the Interrupt Status register will be generated to host CPU to report a bus error condition.

Once the DMAE interrupt is received, the software shall send a CMD12 to abort the current transfer and read the DS\_ADDR bits of the DMA System Address register to get the starting address of the corrupted block. After the DMA error is fixed, the software should apply a data reset and re-start the transfer from this address to recover the corrupted block. DMA operation will resume when the interrupt is serviced by software.



### 54.4.2.4 ADMA Engine

In the SD Host Controller Standard, a new DMA transfer algorithm called the ADMA (Advanced DMA) is defined. For Simple DMA, once the page boundary is reached, a DMA interrupt will be generated and the new system address shall be programmed by the Host Driver.

The ADMA defines the programmable descriptor table in the system memory. The Host Driver can calculate the system address at the page boundary and program the descriptor table before executing ADMA. It reduces the frequency of interrupts to the host system. Therefore, higher speed DMA transfers could be realized since the Host MCU intervention would not be needed during long DMA based data transfers.

There are two types of ADMA: ADMA1 and ADMA2 in Host Controller. ADMA1 can support data transfer of 4KB aligned data in system memory. ADMA2 improves the restriction so that data of any location and any size can be transferred in system memory. Their formats of Descriptor Table are different.

ADMA can recognize all kinds of descriptors define in SD Host Controller Standard, and if 'End' flag is detected in the descriptor, ADMA will stop after this descriptor is processed.

#### 54.4.2.4.1 ADMA Concept and Descriptor Format

For ADMA1, including the following descriptors:

- Valid/Invalid descriptor.
- Nop descriptor.
- Set data length descriptor.
- Set data address descriptor.
- Link descriptor.
- Interrupt flag and End flag in descriptor.

For ADMA2, including the following descriptors:

- Valid/Invalid descriptor.
- Nop descriptor.
- Rsv descriptor.
- Set data length & address descriptor.
- Link descriptor.
- Interrupt flag and End flag in descriptor.

See [Figure 54-8](#) for the format of the descriptor table for ADMA1.

## Functional Description

ADMA2 deals with the lower 32-bit first, and then the higher 32-bit. If the 'Valid' flag of descriptor is 0, it will ignore the high 32-bit. Address field shall be set on word aligned(lower 2-bit is always set to 0). Data length is in byte unit.

ADMA will start read/write operation after it reaches the Tran state, using the data length and data address analyzed from most recent descriptor(s).

For ADMA1, the valid data length descriptor is the last Set type descriptor before Tran type descriptor. Every Tran type will trigger a transfer, and the transfer data length is extracted from the most recent Set type descriptor. If there is no Set type descriptor after the previous Trans descriptor, the data length will be the value for previous transfer, or 0 if no Set descriptor is ever met.

For ADMA2, Tran type descriptor contains both data length and transfer data address, so only a Tran type descriptor can start a data transfer

Address/ Page Field		Address/ Page Field		Attribute Field					
31	12	11	6	5	4	3	2	1	0
Address or Data Length		000000		Act 2	Act 1	0	Int	End	Valid

Act 2	Act1	Symbol	Comment	31- 28	27- 12
0	0	Nop	No Operation	Don't Care	
0	1	Set	Set Data Length	0000	Data Length
1	0	Tran	Transfer Data	Data Address	
1	1	Link	Link Descriptor	Descriptor Address	

Valid	Valid = 1 indicates this line of descriptor is effective. If Valid = 0 generate ADMA Error Interrupt and stop ADMA.
End	End = 1 indicates current descriptor is the ending one.
Int	Int = 1 generates DMA Interrupt when this descriptor is processed.

**Figure 54-8. Format of the ADMA1 Descriptor Table**

### 54.4.2.4.2 ADMA Interrupt

If the interrupt flag descriptor is set, ADMA will generate an interrupt according to various types of descriptors:

For ADMA1:

- Set type of descriptor: interrupt is generated when data length is set.
- Tran type descriptor: interrupt is generated when this transfer is complete.
- Link type of descriptor: interrupt is generated when new descriptor address is set.
- Nop type of descriptor: interrupt is generated just after this descriptor is fetched.

For ADMA2:

- Tran type of descriptor: interrupt is generated when this transfer is complete.
- Link type of descriptor: interrupt is generated when new descriptor address is set.
- Nop/Rsv type of descriptor: interrupt is generated just after this descriptor is fetched.

#### 54.4.2.4.3 ADMA Error

The ADMA will stop whenever any error is encountered. These errors include:

- Fetching descriptor error
- AHB response error
- Data length mismatch error

An ADMA descriptor error will be generated when it fails to detect a 'Valid' flag in the descriptor. If an ADMA descriptor error occurs, the interrupt is not generated even if the 'Interrupt' flag of this descriptor is set.

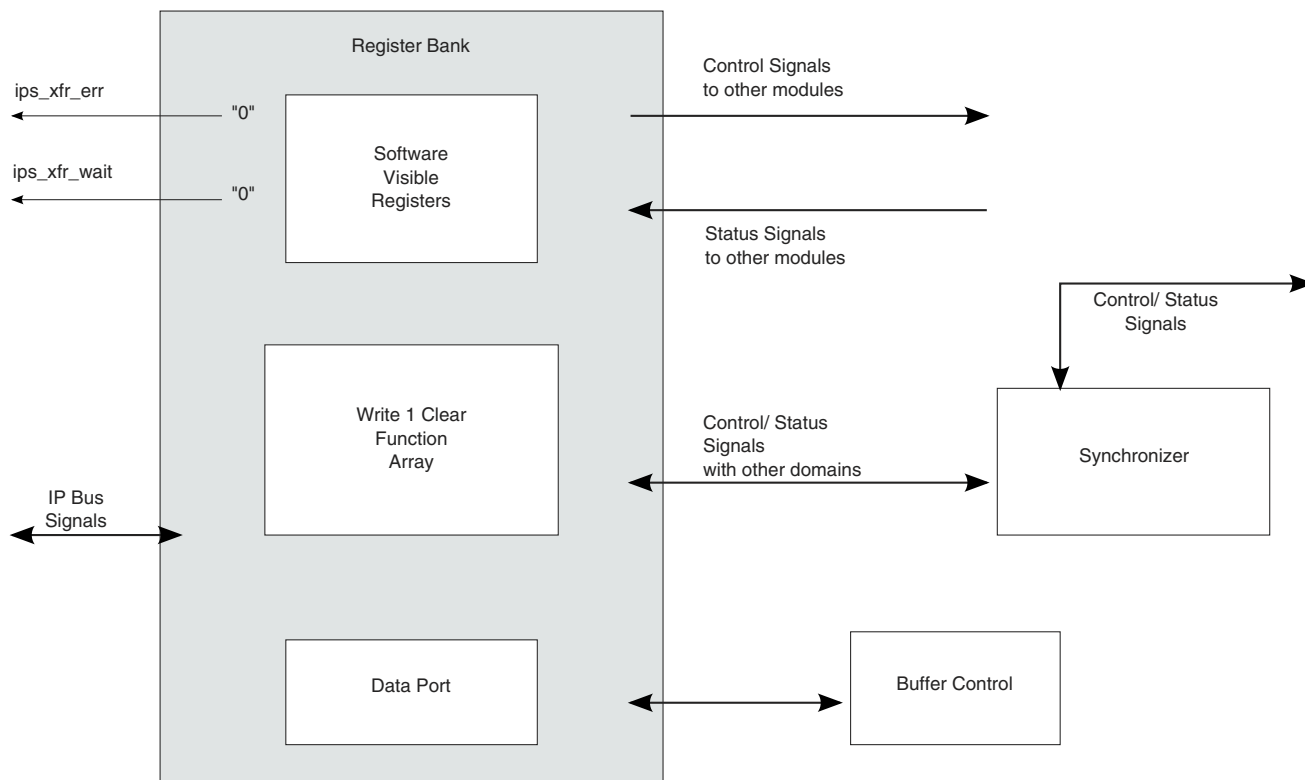
When BLKCNTEN bit is set, data length set in buffer must be equal to the whole data length set in descriptor nodes, otherwise data length mismatch error will be generated.

When BLKCNTEN bit is not set, then whole data length set in descriptor should be a multiple of block lengths; otherwise, when data set in the descriptor nodes are not performed at block boundaries, then data mismatch errors will occur.

### 54.4.3 Register Bank with IP Bus Interface

Register accesses via the IP Bus interface are actually on the Register Bank.

See [Figure 54-9](#) below for the block diagram.



**Figure 54-9. Register Bank Diagram**

Only 32-bit access is allowed, and no partial read / write is supported, thus all accesses are word aligned.

### 54.4.3.1 SD Protocol Unit

The SD protocol unit deals with all SD protocol affairs.

The SD Protocol Unit performs the following functions:

- Acts as the bridge between the internal buffer and the SD bus
- Sends the command data as well as its argument serially
- Stores the serial response bit stream into corresponding registers
- Detects the bus state on the CMD/DAT lines
- Monitors the interrupt from the SDIO card
- Asserts the read wait signal
- Gates off the SD clock when buffer is announcing danger status
- Detects the write protect state

The SD Protocol Unit consists of four sub modules:

1. SD control misc.

2. Command control.
3. Data control.
4. Clock control

#### **54.4.3.2 SD control misc**

In the SD control misc unit, the card detect(include the CD\_B and DATA3 used as Card Detection), write protection and card interrupt are implemented.

This module monitors the signal level on all 8 data lines, the command lines, and directly routes the level values into the Register Bank. The driver can use this for debug purposes.

The module also detects the WP (Write Protect) line. If WP is active, writes to the register bank will be ignored.

This module also drives the LCTL output signal when the LCTL bit is set by the driver.

#### **54.4.3.3 SD Clock control**

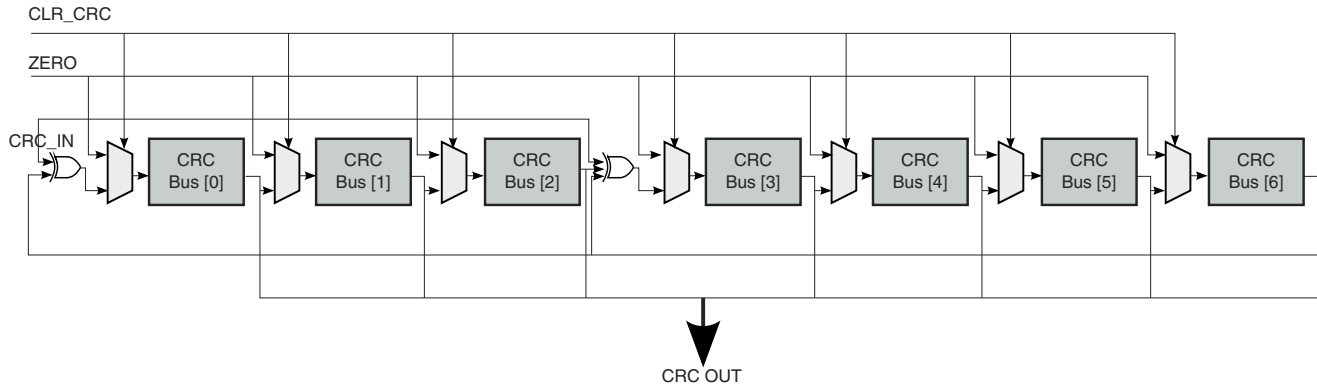
If the internal data buffer is near full(for read) or near empty(for write), the SD clock must be gated off to avoid buffer over/under-run, this module will assert the gate of the output SD clock to shut the clock off. After the buffer has space(for read) or has data(for write), the clock gate of this module will open and the SD clock will be active again.

#### **54.4.3.4 Command control**

The Command Control module deals with the transactions on the CMD line.

See the figure below for an illustration of the structure for the Command CRC Shift Register.

## Functional Description



**Figure 54-10. Command CRC Shift Register**

The CRC polynomials for the CMD are as follows:

Generator polynomial:  $G(x) = x^7 + x^3 + 1$   
 $M(x) = (\text{first bit}) * x^n + (\text{second bit}) * x^{n-1} + \dots + (\text{last bit}) * x^0$   
 $\text{CRC}[6:0] = \text{Remainder} [(M(x) * x^7) / G(x)]$

### 54.4.3.5 Data control

The Data Agent deals with the transactions on the eight data lines. Moreover, this module also detects the busy state on the DATA0 line, and generates the Read Wait state by the request from the Transceiver.

The CRC polynomials for the DATA are as follows:

Generator polynomial:  $G(x) = x^{16} + x^{12} + x^5 + 1$   
 $M(x) = (\text{first bit}) * x^n + (\text{second bit}) * x^{n-1} + \dots + (\text{last bit}) * x^0$   
 $\text{CRC}[15:0] = \text{Remainder} [(M(x) * x^{16}) / G(x)]$

### 54.4.4 Clock & Reset Manager

This module controls all the reset signals within the uSDHC.

There are four kinds of reset signals within uSDHC:

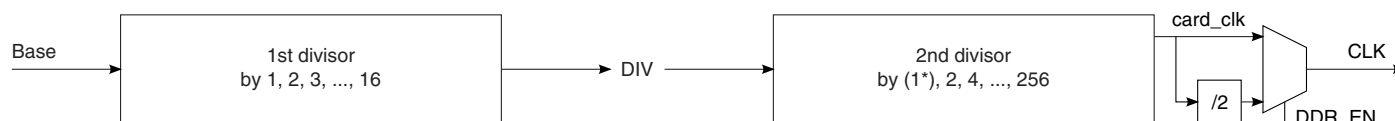
1. Hardware reset.
2. Software reset for all logic.
3. Software reset for the data logic.
4. Software reset for the command logic.

All these signals are fed into this module and stable signals are generated inside the module to reset all other modules. The module also gates off all the inside signals.

### 54.4.5 Clock Generator

The Clock Generator generates the card CLK by peripheral source clock in two stages.

Refer to the figure below for the structure of the divider. The term "Base" represents the frequency of peripheral source clock.



**Figure 54-11. Two Stages of the Clock Divider**

The first stage outputs an intermediate clock (DIV), which can be Base, Base/2, Base/3, ..., or Base/16.

The second stage is a prescaler, and outputs the actual internal working clock (card\_clk). This clock is the driving clock for all sub modules of the SD Protocol Unit, and the sync FIFOs (see [Figure 54-3](#)) to synchronize with the data rate from the internal data buffer. The frequency of the clock output from this stage, can be DIV, DIV/2, DIV/4,..., or DIV/256. Thus the highest frequency of the card\_clk is Base, and the next highest is Base/2, while the lowest frequency is Base/4096. If the duty cycle of Base clock is 50%, the duty cycle of card\_clk is also 50%, even when the compound divisor is an odd value.

Please note, in SDR mode and DDR mode, the CLK are different.

- In SDR mode, CLK is equal to the internal working clock(card\_clk).
- In DDR mode, CLK is equal to the card\_clk/2.

### 54.4.6 SDIO Card Interrupt

#### 54.4.6.1 Interrupts in 1-bit Mode

In this case the DATA1 pin is dedicated to providing the interrupt function. An interrupt is asserted by pulling the DATA1 low from the SDIO card, until the interrupt service is finished to clear the interrupt.

### 54.4.6.2 Interrupt in 4-bit Mode

Since the interrupt and data line 1 share Pin 8 in 4-bit mode, an interrupt will only be sent by the card and recognized by the host during a specific time. This is known as the Interrupt Period. The uSDHC will only sample the level on Pin 8 during the Interrupt Period. At all other times, the host will ignore the level on Pin 8, and treat it as the data signal. The definition of the Interrupt Period is different for operations with single block and multiple block data transfers.

In the case of normal single data block transmissions, the Interrupt Period becomes active two clock cycles after the completion of a data packet. This Interrupt Period lasts until after the card receives the end bit of the next command that has a data block transfer associated with it.

For multiple block data transfers in 4-bit mode, there is only a limited period of time that the Interrupt Period can be active due to the limited period of data line availability between the multiple blocks of data. This requires a more strict definition of the Interrupt Period. For this case, the Interrupt Period is limited to two clock cycles. This begins two clocks after the end bit of the previous data block. During this 2-clock cycle interrupt period, if an interrupt is pending, the DATA1 line will be held low for one clock cycle with the last clock cycle pulling DATA1 high. On completion of the Interrupt Period, the card releases the DATA1 line into the high Z state. The uSDHC samples the DATA1 during the Interrupt Period when the IABG bit in the Protocol Control register is set.

Refer to SDIO Card Specification v1.10f for further information about the SDIO card interrupt.

### 54.4.6.3 Card Interrupt Handling

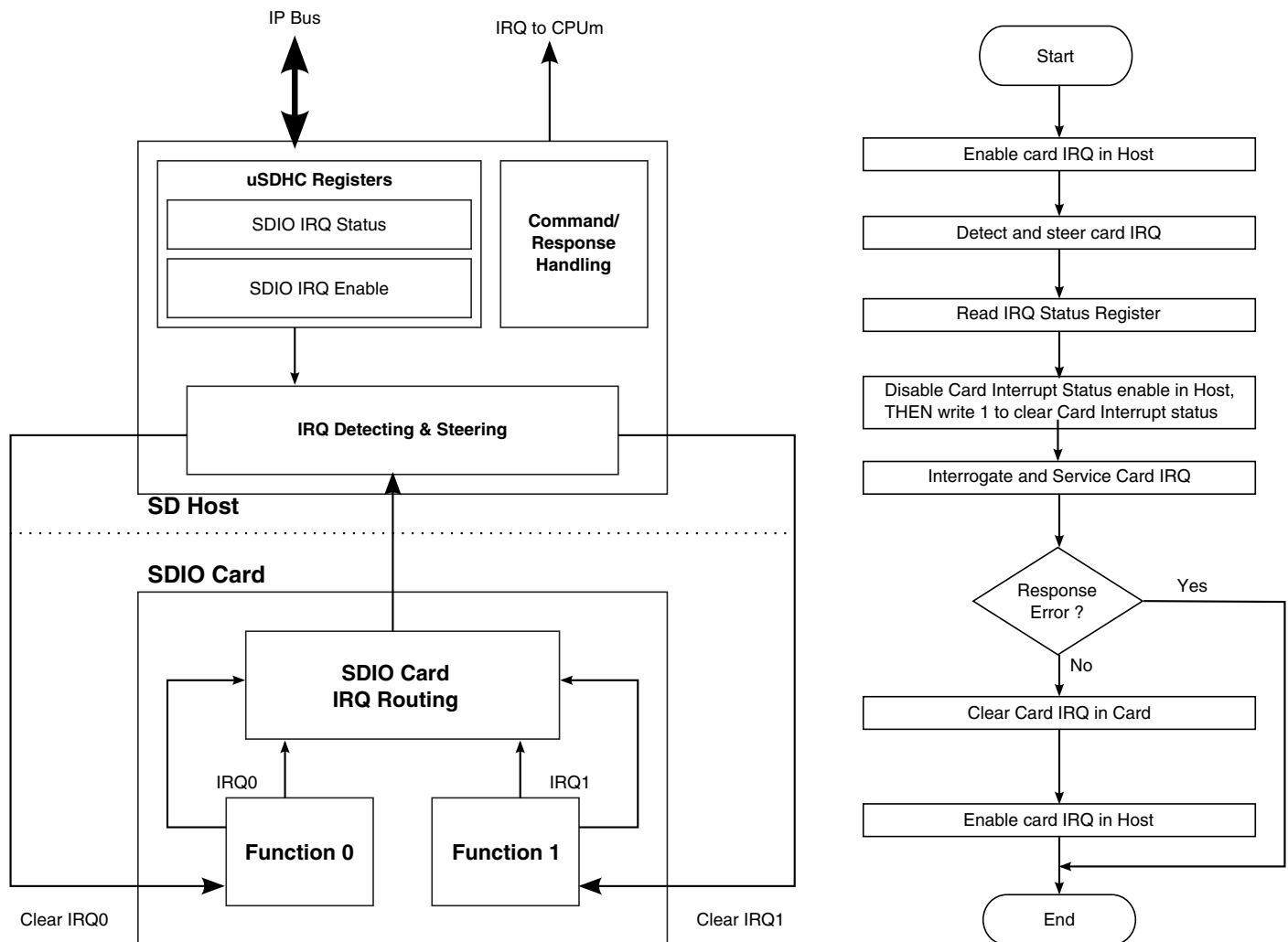
When the CINTIEN bit in the Interrupt Signal Enable Register is set to 0, the uSDHC clears the interrupt request to the Host System. The Host Driver should clear this bit before servicing the SDIO Interrupt and should set this bit again after all interrupt requests from the card are cleared to prevent inadvertent interrupts.

The SDIO Card Interrupt Status can be cleared by writing 1 to this bit. But as the interrupt source from the SDIO card does not clear, this bit is set again. In order to clear this bit, it is required to reset the interrupt source from the external card followed by a writing 1 to this bit. In 1-bit mode, the uSDHC will detect the SDIO Interrupt with or without the SD clock (to support wakeup). In 4-bit mode, the interrupt signal is sampled during the Interrupt Period, so there are some sample delays between the interrupt signal from the SDIO card and the interrupt to the Host System Interrupt Controller. When the SDIO status has been set, and the Host Driver needs to service this interrupt, so the SDIO bit in the Interrupt Control Register of SDIO card will be cleared. This is required to clear



the SDIO interrupt status latched in the uSDHC and to stop driving the interrupt signal to the System Interrupt Controller. The Host Driver must issue a CMD52 to clear the card interrupt. After completion of the card interrupt service, the SDIO Interrupt Status Enable bit is set to 1, and the uSDHC starts sampling the interrupt signal again.

See the figure below for an illustration of the SDIO card interrupt scheme and for the sequences of software and hardware events that take place during a card interrupt handling procedure.



**Figure 54-12. Card Interrupt Scheme and Card Interrupt Detection and Handling Procedure**

### 54.4.7 Card Insertion and Removal Detection

The uSDHC uses either the DATA3 pin or the CD\_B pin to detect card insertion or removal. When there is no card on the MMC/SD bus, the DATA3 will be pulled to a low voltage level by default.

When any card is inserted to or removed from the socket, the uSDHC detects the logic value changes on the DATA3 pin and generates an interrupt. When the DATA3 pin is not used for card detection (for example, it is implemented in GPIO), the CD\_B pin must be connected for card detection. Whether DATA3 is configured for card detection or not, the CD\_B pin is always a reference for card detection. Whether the DATA3 pin or the CD\_B pin is used to detect card insertion, the uSDHC will send an interrupt (if enabled) to inform the Host system that a card is inserted.

### 54.4.8 Power Management and Wake Up Events

When there is no operation between the uSDHC and the card through the SD bus, the user can completely disable the ipg\_clk and ipg\_perclk in the chip level clock control module to save power. When the user needs to use the uSDHC to communicate with the card, it can enable the clock and start the operation.

In some circumstances, when the clocks to the uSDHC are disabled, for instance, when the system is in low power mode, there are some events for which the user needs to enable the clock and handle the event. These events are called wakeup interrupts. The uSDHC can generate these interrupt even when there are no clocks enabled. The three interrupts which can be used as wake up events are:

1. Card Removal Interrupt
2. Card Insertion Interrupt
3. Interrupt from SDIO card

The uSDHC offers a power management feature.

By clearing the clock enabled bits in the System Control Register, the clocks are gated in the low position to the uSDHC. For maximum power saving, the user can disable all the clocks to the uSDHC when there is no operation in progress.

These three wake up events (or wakeup interrupts) can also be used to wake up the system from low-power modes.

#### NOTE

To make the interrupt a wakeup event, when all the clocks to the uSDHC are disabled or when the whole system is in low power mode, the corresponding wakeup enabled bit needs to be

set. Refer to [Protocol Control \(uSDHC\\_PROT\\_CTRL\)](#) for more information on the uSDHC Protocol Control register.

### 54.4.8.1 Setting Wake Up Events

For the uSDHC to respond to a wakeup event, the software must set the respective wakeup enable bit before the CPU enters sleep mode.

Before the software disables the host clock, it should ensure that all of the following conditions have been met:

- No Read or Write Transfer is active
- Data and Command lines are not active
- No interrupts are pending
- Internal data buffer is empty

## 54.4.9 MMC fast boot

The Embedded MultiMediaCard (eMMC4.3) specification adds a fast boot feature which requires hardware support.

In boot operation mode, the master (MultiMediaCard host) can read boot data from the slave (MMC device) by keeping CMD line low after power-on, or sending CMD0 with argument + 0xFFFFFFFFFA (optional for slave), before issuing CMD1.

There are two types of fast boot mode, boot operation and alternative boot operation, in the eMMC4.3 specification. Each type also has with-acknowledge and without-acknowledge modes.

### NOTE

For the eMMC4.3 card setting, please see the eMMC4.3 specification.

### 54.4.9.1 Boot operation

### NOTE

For the purposes of this documentation, fast boot is called "normal fast boot mode".

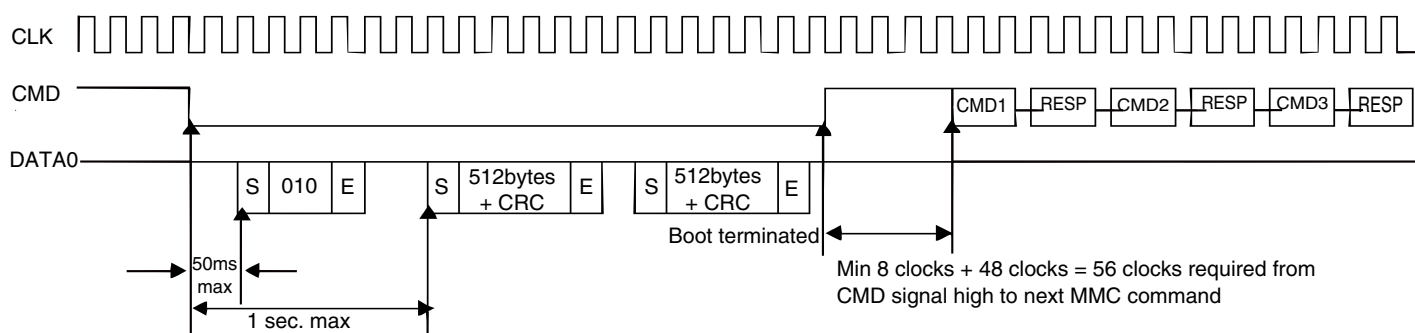
If the CMD line is held LOW for 74 clock cycles and more after power-up before the first command is issued, the slave recognizes that boot mode is being initiated and starts preparing boot data internally.

Within 1 second after the CMD line goes LOW, the slave starts to send the first boot data to the master on the DATA line(s). The master must keep the CMD line LOW to read all of the boot data.

If boot acknowledge is enabled, the slave has to send acknowledge pattern '010' to the master within 50ms after the CMD line goes LOW. If boot acknowledge is disabled, the slave will not send out acknowledge pattern '010'.

The master can terminate boot mode with the CMD line HIGH.

Boot operation will be terminated when all contents of the enabled boot data are sent to the master. After boot operation is executed, the slave shall be ready for CMD1 operation and the master needs to start a normal MMC initialization sequence by sending CMD1.



**Figure 54-13. MultiMediaCard state diagram (normal boot mode)**

## 54.4.9.2 Alternative boot operation

This boot function is optional for the device. If bit 0 in the extended CSD byte[228] is set to '1', the device supports the alternative boot operation.

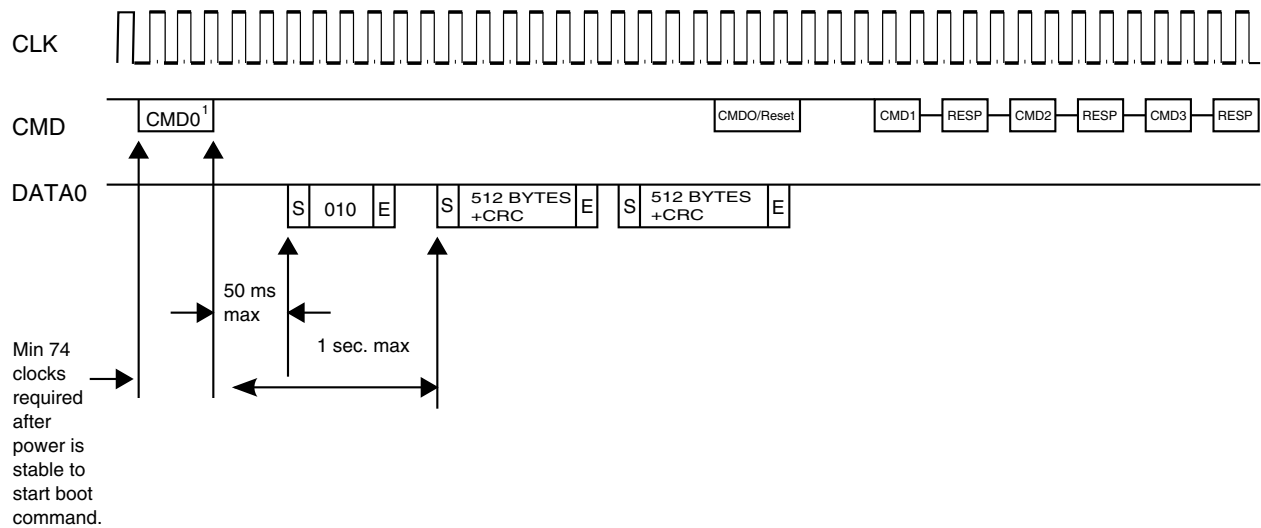
After power-up, if the host issues CMD0 with the argument of 0xFFFFFFFFFA after 74 clock cycles, before CMD1 is issued or the CMD line goes low, the slave recognizes that boot mode is being initiated and starts preparing boot data internally.

Within 1 second after CMD0 with the argument of 0xFFFFFFFFFA is issued, the slave starts to send the first boot data to the master on the DATA line(s).

If boot acknowledge is enabled, the slave has to send the acknowledge pattern '010' to the master within 50ms after the CMD0 with the argument of 0xFFFFFFFFFA is received. If boot acknowledge is disabled, the slave will not send out acknowledge pattern '010'.

The master can terminate boot mode by issuing CMD0 (Reset).

Boot operation will be terminated when all contents of the enabled boot data are sent to the master. After boot operation is executed, the slave shall be ready for CMD1 operation and the master needs to start a normal MMC initialization sequence by sending CMD1.



NOTE 1. CMD0 with argument 0xFFFFFFFF

**Figure 54-14. MultiMediaCard state diagram (alternative boot mode)**

## 54.5 Initialization/Application of uSDHC

All communication between the system and cards are controlled by the host. The host sends commands of two types: broadcast and addressed (point-to-point).

Broadcast commands are intended for all cards, such as GO\_IDLE\_STATE, SEND\_OP\_COND, ALL\_SEND\_CID. In Broadcast mode, all cards are in the open-drain mode to avoid bus contention. Refer to [Commands for MMC/SD/SDIO](#) for the commands of bc and bcr categories.

After the Broadcast command CMD3 is issued, the cards enter standby mode. Addressed type commands are used from this point. In this mode, the CMD/DATA I/O pads will turn to push-pull mode, to have the driving capability for maximum frequency operation. Refer to [Commands for MMC/SD/SDIO](#) for the commands of ac and adtc categories.

## 54.5.1 Command Send & Response Receive Basic Operation

Assuming the data type WORD is an unsigned 32-bit integer, the below flow is a guideline for sending a command to the card(s):

```
send_command(cmd_index, cmd_arg, other requirements)
{
    WORD wCmd; // 32-bit integer to make up the data to write into Transfer Type register, it is
    recommended to implement in a bit-field manner
    wCmd = (<cmd_index> & 0x3f) >> 24; // set the first 8 bits as '00'+<cmd_index>
    set CMDTYP, DPSEL, CICCEN, CCCEN, RSTTYP, DTDSEL accorind to the command_index;
    if (internal DMA is used) wCmd |= 0x1;
    if (multi-block transfer) {
        set MSBSEL bit;
        if (finite block number) {
            set BCEN bit;
            if (auto12 command is to use) set AC12EN bit;
        }
    }
    write_reg(CMDARG, <cmd_arg>); // configure the command argument
    write_reg(XFERTYP, wCmd); // set Transfer Type register as wCmd value to issue the command
}
wait_for_response(cmd_index)
{
    while (CC bit in IRQ Status register is not set); // wait until Command Complete bit is set
    read IRQ Status register and check if any error bits about Command are set
    if (any error bits are set) report error;
    write 1 to clear CC bit and all Command Error bits;
}
```

For the sake of simplicity, the function wait\_for\_response is implemented here by means of polling. For an effective and formal way, the response is usually checked after the Command Complete Interrupt is received. When doing this, make sure the corresponding interrupt status bits are enabled.

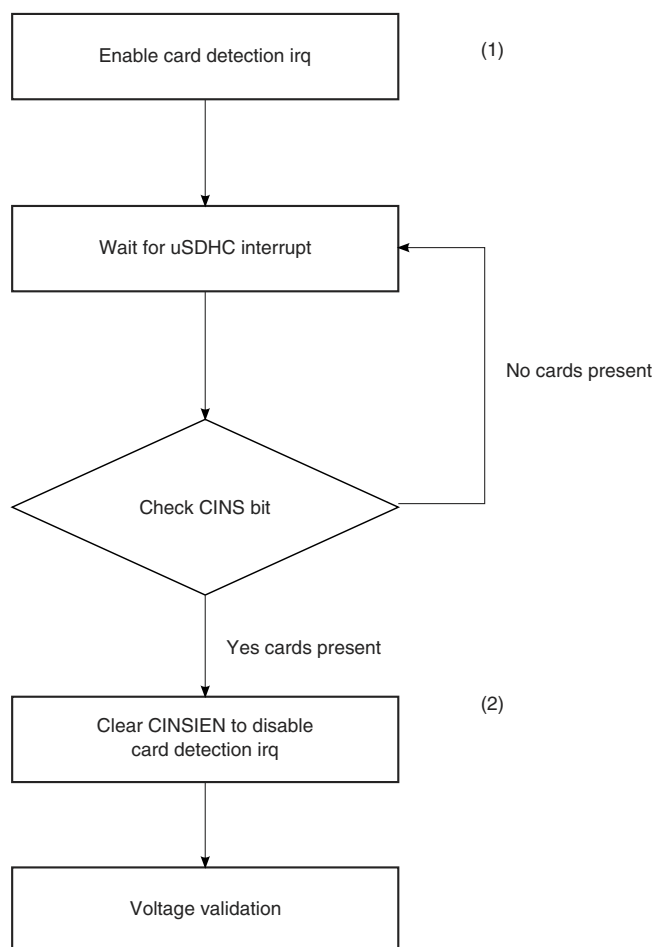
For some scenarios, the response time-out is expected. For instance, after all cards respond to CMD3 and go to the Standby State, no response to the Host when CMD2 is sent. The Host Driver will deal with "fake" errors like this with caution.

## 54.5.2 Card Identification Mode

When a card is inserted to the socket or the card was reset by the host, the host needs to validate the operation voltage range, identify the cards, request the cards to publish the Relative Card Address (RCA) or to set the RCA for the MMC cards.

### 54.5.2.1 Card Detect

See the figure below for a flow diagram showing the detection of MMC, SD and SDIO cards using the uSDHC.



**Figure 54-15. Flow Diagram for Card Detection**

Here is the card detect sequence:

- Set the CINSIEN bit to enable card detection interrupt
- When an interrupt from the uSDHC is received, check the CINS bit in the Interrupt Status register to see if it was caused by card insertion
- Clear the CINSIEN bit to disable the card detection interrupt and ignore all card insertion interrupts afterwards

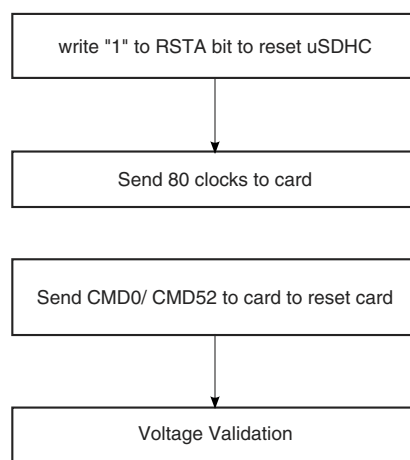
### 54.5.2.2 Reset

The host consists of three types of resets:

- Hardware reset (Card and Host) which is driven by POR (Power On Reset)

- Software reset (Host Only) is initiated by the write operation on the RSTD, RSTC, or RSTA bits of the System Control register to reset the data part, command part, or all parts of the Host Controller, respectively
- Card reset (Card Only). The command, "Go\_Idle\_State" (CMD0), is the software reset command for all types of MMC cards, SD Memory cards. This command sets each card into the Idle State regardless of the current card state. For an SD I/O Card, CMD52 is used to write an I/O reset in the CCCR. The cards are initialized with a default relative card address (RCA=0x0000) and with a default driver stage register setting (lowest speed, highest driving current capability).

After the card is reset, the host needs to validate the voltage range of the card. See the figure below for the software flow to reset both the uSDHC and the card.



**Figure 54-16. Flow Chart for Reset of the uSDHC and SD I/O Card**

```

software_reset()
{
  set_bit(SYSCTRL, RSTA); // software reset the Host
  set DTOCV and SDCLKFS bit fields to get the CLK of frequency around 400kHz
  configure IO pad to set the power voltage of external card to around 3.0V
  poll bits CIHB and CDIHB bits of PRSSTAT to wait both bits are cleared
  set_bit(SYSCTRL, INTIA); // send 80 clock ticks for card to power up
  send_command(CMD_GO_IDLE_STATE, <other parameters>); // reset the card with CMD0
  or send_command(CMD_IO_RW_DIRECT, <other parameters>);
}
  
```

### 54.5.2.3 Voltage Validation

All cards should be able to establish communication with the host using any operation voltage in the maximum allowed voltage range specified in the card specification. However, the supported minimum and maximum values for V<sub>dd</sub> are defined in the Operation Conditions Register (OCR) and may not cover the whole range.



Cards that store the CID and CSD data in the preload memory are only able to communicate this information under data transfer Vdd conditions. This means if the host and card have non-common Vdd ranges, the card will not be able to complete the identification cycle, nor will it be able to send CSD data.

Therefore, a special command Send\_Op\_Cont (CMD1 for MMC), SD\_Send\_Op\_Cont (ACMD41 for SD Memory) and IO\_Send\_Op\_Cont (CMD5 for SD I/O) is used. The voltage validation procedure is designed to provide a mechanism to identify and reject cards which do not match the Vdd range(s) desired by the host. This is accomplished by the host sending the desired Vdd voltage window as the operand of this command. Cards that can't perform the data transfer in the specified range must discard themselves from further bus operations and go into the Inactive State. By omitting the voltage range in the command, the host can query each card and determine the common voltage range before sending out-of-range cards into the Inactive State. This query should be used if the host is able to select a common voltage range or if a notification shall be sent to the system when a non-usable card in the stack is detected.

The following steps show how to perform voltage validation when a card is inserted:

```

voltage_validation(voltage_range_argument)
{
    label the card as UNKNOWN;
    send_command(IO_SEND_OP_COND, 0x0, <other parameters are omitted>); // CMD5, check SDIO
    operation voltage, command argument is zero
    if (RESP_TIMEOUT != wait_for_response(IO_SEND_OP_COND)) { // SDIO command is accepted
        if (0 < number of IO functions) {
            label the card as SDIO;
            IORDY = 0;
            while (!(IORDY in IO OCR response)) { // set voltage range for each IO
                function
                    send_command(IO_SEND_OP_COND, <voltage range>, <other
parameter>);
                    wait_for_response(IO_SEND_OP_COND);
            } // end of while ...
        } // end of if (0 < ...
        if (memory part is present inside SDIO card) Label the card as SDCCombo; // this is
an
SD-Combo card
    } // end of if (RESP_TIMEOUT ...
    if (the card is labelled as SDIO card) return; // card type is identified and voltage range
is
set, so exit the function;
    send_command(APP_CMD, 0x0, <other parameters are omitted>); // CMD55, Application specific
CMD
prefix
    if (no error calling wait_for_response(APP_CMD, <...>)) { // CMD55 is accepted
        send_command(SD_APP_OP_COND, <voltage range>, <...>); // ACMD41, to set voltage
range
        for memory part or SD card
            wait_for_response(SD_APP_OP_COND); // voltage range is set
            if (card type is UNKNOWN) label the card as SD;
            return; //
    } // end of if (no error ...
    else if (errors other than time-out occur) { // command/response pair is corrupted
        deal with it by program specific manner;
    } // of else if (response time-out
    else { // CMD55 is refuse, it must be MMC card if (card is already labelled as SDCCombo)
    { //
change label

```

```

        re-label the card as SDIO;
        ignore the error or report it;
        return; // card is identified as SDIO card
    } // of if (card is ...
    send_command(SEND_OP_COND, <voltage range>, <...>);
    if (RESP_TIMEOUT == wait_for_response(SEND_OP_COND)) { // CMD1 is not accepted,
either
        label the card as UNKNOWN;
        return;
    } // of if (RESP_TIMEOUT ...
} // of else
}

```

### 54.5.2.4 Card Registry

Card registry for the MMC and SD/SDIO/SD Combo cards are different. For the SD Card, the Identification process starts at a clock rate lower than 400 kHz and the power voltage higher than 2.7 V (as defined by the Card spec). At this time, the CMD line output drives are push-pull drivers instead of open-drain. After the bus is activated, the host will request the card to send their valid operation conditions.

The response to ACMD41 is the operation condition register of the card. The same command shall be send to all of the new cards in the system. Incompatible cards are put into the Inactive State. The host then issues the command, All\_Send\_CID (CMD2), to each card to get its unique card identification (CID) number. Cards that are currently unidentified (in the Ready State), send their CID number as the response. After the CID is sent by the card, the card goes into the Identification State.

The host then issues Send\_Relative\_Addr (CMD3), requesting the card to publish a new relative card address (RCA) that is shorter than the CID. This RCA will be used to address the card for future data transfer operations. Once the RCA is received, the card changes its state to the Standby State. At this point, if the host wants the card to have an alternative RCA number, it may ask the card to publish a new number by sending another Send\_Relative\_Addr command to the card. The last published RCA is the actual RCA of the card.

The host repeats the identification process with CMD2 and CMD3 for each card in the system until the last CMD2 gets no response from any of the cards in system.

For MMC operation, the host starts the card identification process in open-drain mode with the identification clock rate lower than 400 kHz and the power voltage higher than 2.7 V. The open drain driver stages on the CMD line allow parallel card operation during card identification. After the bus is activated the host will request the cards to send their valid operation conditions (CMD1). The response to CMD1 is the "wired OR" operation on the condition restrictions of all cards in the system. Incompatible cards are sent into the Inactive State. The host then issues the broadcast command All\_Send\_CID (CMD2), asking all cards for their unique card identification (CID) number. All unidentified cards

(the cards in Ready State) simultaneously start sending their CID numbers serially, while bit-wise monitoring their outgoing bit stream. Those cards, whose outgoing CID bits do not match the corresponding bits on the command line in any one of the bit periods, stop sending their CID immediately and must wait for the next identification cycle. Since the CID is unique for each card, only one card can be successfully send its full CID to the host. This card then goes into the Identification State. Thereafter, the host issues Set\_Relative\_Addr (CMD3) to assign to the card a relative card address (RCA). Once the RCA is received the card state changes to the Stand-by State, and the card does not react in further identification cycles, and its output driver switches from open-drain to push-pull. The host repeats the process, mainly CMD2 and CMD3, until the host receives a time-out condition to recognize the completion of the identification process.

For operation as MMC cards:

```
card_registry()
{
do { // decide RCA for each card until response time-out
    if(card is labelled as SDCCombo or SDIO) { // for SDIO card like device
        send_command(SET_RELATIVE_ADDR, 0x00, <...>); // ask SDIO card to
publish its
RCA
        retrieve RCA from response;
    } // end if (card is labelled as SDCCombo ...
    else if (card is labelled as SD) { // for SD card
        send_command(ALL_SEND_CID, <...>);
        if (RESP_TIMEOUT == wait_for_response(ALL_SEND_CID)) break;
        send_command(SET_RELATIVE_ADDR, <...>);
        retrieve RCA from response;
    } // else if (card is labelled as SD ...
    else if (card is labelled as MMC) {
        send_command(ALL_SEND_CID, <...>);
        rca = 0x1; // arbitrarily set RCA, 1 here for example
        send_command(SET_RELATIVE_ADDR, 0x1 << 16, <...>); // send RCA at upper
16
bits
        } // end of else if (card is labelled as MMC ...
    } while (response is not time-out);
}
```

## 54.5.3 Card Access

### 54.5.3.1 Block Write

#### 54.5.3.1.1 Normal Write

During a block write (CMD24 - 27, CMD60, CMD61), one or more blocks of data are transferred from the host to the card with a CRC appended to the end of each block by the host. If the CRC fails, the card shall indicate the failure on the DATA line. The transferred data will be discarded and not written, and all further transmitted blocks (in multiple block write mode) will be ignored.

If the host uses partial blocks whose accumulated length is not block aligned and block misalignment is not allowed (CSD parameter WRITE\_BLK\_MISALIGN is not set), the card detects the block misalignment error and aborts the programming before the beginning of the first misaligned block. The card sets the ADDRESS\_ERROR error bit in the status register, and while ignoring all further data transfer, waits in the Receive-data-State for a stop command. The write operation is also aborted if the host tries to write over a write protected area.

For MMC and SD cards, programming of the CID and CSD registers does not require a previous block length setting. The transferred data is also CRC protected. If a part of the CSD or CID register is stored in ROM, then this unchangeable part must match the corresponding part of the receive buffer. If this match fails, then the card will report an error and not change any register contents.

For all types of cards, some may require long and unpredictable periods of time to write a block of data. After receiving a block of data and completing the CRC check, the card will begin writing and hold the DATA line low if its write buffer is full and unable to accept new data from a new WRITE\_BLOCK command. The host may poll the status of the card with a SEND\_STATUS command (CMD13) or other means for SDIO cards at any time, and the card will respond with its status. The responded status indicates whether the card can accept new data or whether the write process is still in progress. The host may deselect the card by issuing a CMD7 (to select a different card) to place the card into the Standby State and release the DATA line without interrupting the write operation. When re-selecting the card, it will reactivate the busy indication by pulling DATA to low if the programming is still in progress and the write buffer is unavailable.

The software flow to write to a card incorporates the internal DMA and the write operation is a multi-block write with the Auto CMD12 enabled. For the other two methods (by means of external DMA or CPU polling status) with different transfer methods, the internal DMA parts should be removed and the alternative steps should be straightforward.

The software flow to write to a card is described below:

1. Check the card status, wait until the card is ready for data.
2. Set the card block length/size:
  - For SD/MMC cards, use SET\_BLOCKLEN (CMD16)
  - For SDIO cards or the I/O portion of SDCombo cards, use IO\_RW\_DIRECT (CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7)
3. Set the uSDHC block length register to be the same as the block length set for the card in Step 2.
4. Set the uSDHC number block register (NOB), nob is 5 (for instance).

5. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a write CRC error occurred, or some another error, that occurred during the auto12 command sending and response receiving.

### 54.5.3.1.2 DDR Write

uSDHC supports dual data rate mode.

The software flow to write to a card in ddr mode is described as below:

1. Check the card status, wait until the card is ready for data.
2. For eMMC4.4 card, block length only can be set to 512byte.
3. Set the uSDHC number block register (NOB), nob is 5 (for instance).
4. Set eMMC4.4 card to high speed mode, use SWITCH(CMD6).
5. Set eMMC4.4 card bus with (4-bit /8-bit ddr mode), use SWITCH(CMD6).
6. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The DDR\_EN bit should be set. The AC12EN bit should also be set.
7. Wait for the Transfer Complete interrupt.
8. Check the status bit to see if a write CRC error occurred, or some another error, that occurred during the auto12 command sending and response receiving.

### 54.5.3.1.3 Write with Pause

The write operation can be paused during the transfer. Instead of stopping the CLK at any time to pause all the operations, which is also inaccessible to the Host Driver, the Driver can set the Stop At Block Gap Request(SABGREQ) bit in the Protocol Control register to pause the transfer between the data blocks. As there is no time-out condition in a write operation during the data blocks, a write to all types of cards can be paused in this way, and if the DATA0 line is not required to de-assert to release the busy state, no suspend command is needed.

Like in the flow described in [Normal Write](#), the write with pause is shown with the same kind of write operation:

1. Check the card status, wait until card is ready for data.
2. Set the card block length/size:

- For SD/MMC, use SET\_BLOCKLEN (CMD16)
  - For SDIO cards or the I/O portion of SDCCombo cards, use IO\_RW\_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7)
3. Set the uSDHC block length register to be the same as the block length set for the card in Step 2.
  4. Set the uSDHC number block register (NOB), nob is 5 (for instance).
  5. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
  6. Set the SABGREQ bit.
  7. Wait for the Transfer Complete interrupt.
  8. Clear the SABGREQ bit.
  9. Check the status bit to see if a write CRC error occurred.
  10. Set the CREQ bit to continue the write operation.
  11. Wait for the Transfer Complete interrupt.
  12. Check the status bit to see if a write CRC error occurred, or some another error, that occurred during the auto12 command sending and response receiving.

The number of blocks left during the data transfer is accessible by reading the contents of the BLKCNT field in the Block Attribute register. As the data transfer and the setting of the SABGREQ bit are concurrent, and the delay of register read and the register setting, the actual number of blocks left may not be exactly the value read earlier. The Driver shall read the value of BLKCNT after the transfer is paused and the Transfer Complete interrupt is received.

It is also possible the last block has begun when the Stop At Block Gap Request is sent to the buffer. In this case, the next block gap is actually the end of the transfer. These types of requests are ignored and the Driver should treat this as a non-pause transfer and deal with it as a common write operation.

When the write operation is paused, the data transfer inside the Host System is not stopped, and the transfer is active until the data buffer is full. Because of this (if not needed), it is recommended to avoid using the Suspend Command for the SDIO card. This is because when such a command is sent, the uSDHC thinks the System will switch to another function on the SDIO card, and flush the data buffer. The uSDHC takes the Resume Command as a normal command with data transfer, and it is left for the Driver to set all the relevant registers before the transfer is resumed. If there is only one block to send when the transfer is resumed, the MSBSEL and BCEN bits of the Transfer Type register are set as well as the AC12EN bit. However, the uSDHC will automatically send a CMD12 to mark the end of the multi-block transfer.

### 54.5.3.2 Block Read

#### 54.5.3.2.1 Normal Read

For block reads, the basic unit of data transfer is a block whose maximum size is stored in areas defined by the corresponding card specification. A CRC is appended to the end of each block, ensuring data transfer integrity. The CMD17, CMD18, CMD53, CMD60, CMD61, and so on, can initiate a block read. After completing the transfer, the card returns to the Transfer State. For multi blocks read, data blocks will be continuously transferred until a stop command is issued.

The software flow to read from a card incorporates the internal DMA and the read operation is a multi-block read with the Auto CMD12 enabled. For the other two methods (by means of external DMA or CPU polling status) with different transfer methods, the internal DMA parts should be removed and the alternative steps should be straightforward.

The software flow to read from a card is described below:

1. Check the card status, wait until card is ready for data.
2. Set the card block length/size:
  - For SD/MMC, use SET\_BLOCKLEN (CMD16)
  - For SDIO cards or the I/O portion of SDCombo cards, use IO\_RW\_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7)
3. Set the uSDHC block length register to be the same as the block length set for the card in Step 2.
4. Set the uSDHC number block register (NOB), nob is 5 (for instance).
5. Disable the buffer read ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set.
6. Wait for the Transfer Complete interrupt.
7. Check the status bit to see if a read CRC error occurred, or some another error, occurred during the auto12 command sending and response receiving.

#### 54.5.3.2.2 DDR Read

uSDHC supports dual data rate mode.

The software flow to write to a card in ddr mode is described below:

1. Check the card status, wait until the card is ready for data.

2. For eMMC4.4 card, block length only can be set to 512byte.
3. Set the uSDHC number block register (NOB), nob is 5 (for instance).
4. Set eMMC4.4 card to high speed mode, use SWITCH(CMD6).
5. Set eMMC4.4 card bus with (4-bit /8-bit ddr mode), use SWITCH(CMD6).
6. Disable the buffer write ready interrupt, configure the DMA settings and enable the uSDHC DMA when sending the command with data transfer. The DDR\_EN bit should be set. The AC12EN bit should also be set.
7. Wait for the Transfer Complete interrupt.
8. Check the status bit to see if a write CRC error occurred, or some another error, that occurred during the auto12 command sending and response receiving.

### 54.5.3.2.3 Read with Pause

The read operation is not generally able to pause. Only the SDIO card (and SDCombo card working under I/O mode) supporting the Read Wait feature can pause during the read operation. If the SDIO card supports Read Wait (SRW bit in CCCR register is 1), the Driver can set the SABGREQ bit in the Protocol Control register to pause the transfer between the data blocks.

Before setting the SABGREQ bit, make sure the RWCTL bit in the Protocol Control register is set, otherwise the uSDHC will not assert the Read Wait signal during the block gap and data corruption occurs. It is recommended to set the RWCTL bit once the Read Wait capability of the SDIO card is recognized.

Like in the flow described in [Normal Read](#), the read with pause is shown with the same kind of read operation:

1. Check the SRW bit in the CCR register on the SDIO card to confirm the card supports Read Wait.
2. Set the RWCTL bit.
3. Check the card status and wait until the card is ready for data.
4. Set the card block length/size:
  - For SD/MMC, use SET\_BLOCKLEN (CMD16)
  - For SDIO cards or the I/O portion of SDCombo cards, use IO\_RW\_DIRECT(CMD52) to set the I/O Block Size bit field in the CCCR register (for function 0) or FBR register (for functions 1~7)
5. Set the uSDHC block length register to be the same as the block length set for the card in Step 2.
6. Set the uSDHC number block register (NOB), nob is 5 (for instance).
7. Disable the buffer read ready interrupt, configure the DMA setting and enable the uSDHC DMA when sending the command with data transfer. The AC12EN bit should also be set
8. Set the SABGREQ bit.



9. Wait for the Transfer Complete interrupt.
10. Clear the SABGREQ bit.
11. Check the status bit to see if read CRC error occurred.
12. Set the CREQ bit to continue the read operation.
13. Wait for the Transfer Complete interrupt.
14. Check the status bit to see if a read CRC error occurred, or some another error, occurred during the auto12 command sending and response receiving.

Like the write operation, it is possible to meet the ending block of the transfer when paused. In this case, the uSDHC will ignore the Stop At Block Gap Request and treat it as a command read operation.

Unlike the write operation, there is no remaining data inside the buffer when the transfer is paused. All data received before the pause will be transferred to the Host System. No matter if the Suspend Command is sent or not, the internal data buffer is not flushed.

If the Suspend Command is sent and the transfer is later resumed by means of a Resume Command, the uSDHC takes the command as a normal one accompanied with data transfer. It is left for the Driver to set all the relevant registers before the transfer is resumed. If there is only one block to send when the transfer is resumed, the MSBSEL and BCEN bits of the Transfer Type register are set, as well as the AC12EN bit. However, the uSDHC will automatically send the CMD12 to mark the end of multi-block transfer.

#### 54.5.3.2.4 DLL (Delay Line) in Read Path

The DLL(Delay Line) is newly added to assist in sampling read data. The DLL provides the ability to programmatically select a quantized delay (in fractions of the clock period) regardless of on-chip variations such as process, voltage and temperature (PVT).

The reasons why the DLL is needed for uSDHC are 1.) the path of read data traveling from card to host varies. 2.) in SD/MMC DDR mode the minimum input setup and hold time are both at 2.5 ns. The data sampling window is so small that the delay of loopback clock needs to be accurate and consistent regardless of PVT. The DLL takes the divided card\_clk as the reference clock and loopback clock as the input clock. It then generates a delayed version of the input clock according to the programmed target delay.

The DLL can be disabled or bypassed, and it can also be manually set for a fixed delay in override mode. The override value set is the number of delay cells. In override mode, there is no need to set the DLL\_enable. Another DLL mode is target value mode. In this mode, the DLL will automatically adjust the number of delay cells according to target value set by user and PVT changes. Be aware that target value is in units of 1/32 of the

clock reference period. If the card\_clk is 100Mhz, then the reference clock period is 10ns, setting target vaule of 16 means  $5\text{ns} = (16/32) * 10\text{ns}$ . Software can disable automatic update by setting dll\_gate\_update bit. Please refer to [Figure 54-17](#).

Since the user may change the frequency of the card\_clk from time to time by changing SDCLKFS[7:0]/DVS[3:0], the software must adjust the delay value to ensure it works correctly when the reference clock (card\_clk) is changed. The following is the correct flow, which should be ignored if DLL\_CTRL\_ENABLE is not set.

Step 1: Set DLL\_CTRL\_RESET bit

Step 2: Configure the SDCLKFS[7:0] and DVS[3:0]

Step 3: Wait until SDSTB is asserted

Step 4: clear DLL\_CTRL\_RESET bit

Step 5: Wait until both DLL\_STS\_SLV\_LOCK and DLL\_STS\_REF\_LOCK are asserted

Step 6: set DLL\_CTRL\_SLV\_FORCE\_UPD

Step 7: clear DLL\_CTRL\_SLV\_FORCE\_UPD

NOTE:Software should make sure the DLL\_CTRL\_SLV\_FORCE\_UPD is lasted for at least one card\_clk. So software may need to add some delay between step6 and step7.

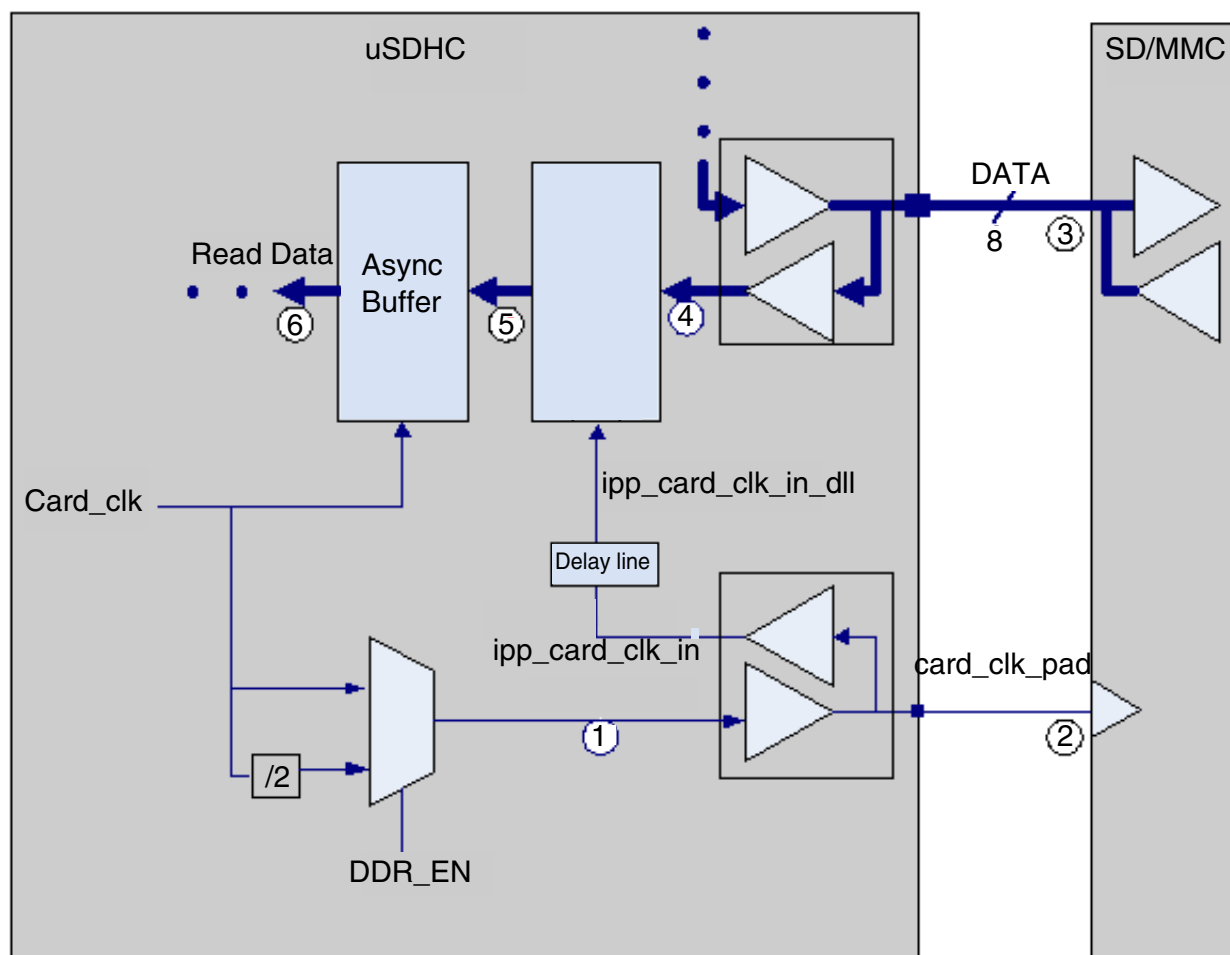


Figure 54-17. DLL(Delay Line) in Read Path

### 54.5.3.3 Suspend Resume

The uSDHC supports the Suspend Resume operations of SDIO cards, although slightly differently than the suggested implementation of Suspend in the SDIO card specification.

#### 54.5.3.3.1 Suspend

After setting the SABGREQ bit, the Host Driver may send a Suspend command to switch to another function of the SDIO card. The uSDHC does not monitor the content of the response, so it doesn't know if the Suspend command succeeded or not. Accordingly, it doesn't de-assert Read Wait for read pause. To solve this problem, the Driver shall not mark the Suspend command as a "Suspend", (i.e. setting the CMDTYP bits to 01). Instead, the Driver shall send this command as if it were a normal command, and only

when the command succeeds, and the BS bit is set in the response, can the Driver send another command marked as "Suspend" to inform the uSDHC that the current transfer is suspended. As shown in the following sequence for Suspend operation:

1. Set the SABREQ bit to pause the current data transfer at block gap.
2. After the BGE bit is set, send the Suspend command to suspend the active function. The CMDTYP bit field must be 2'b00.
3. Check the BS bit of the CCCR in the response. If it is 1, repeat this step until the BS bit is cleared or abandon the suspend operation according to the Driver strategy.
4. Send another normal I/O command to the suspended function. The CMDTYP of this command must be 2'b01, so the uSDHC can detect this special setting and be informed that the paused operation has successfully suspended. If the paused transfer is a read operation, the uSDHC stops driving DATA2 and goes to the idle state.
5. Save the context registers in the system memory for later use, including the DMA System Address Register (for internal DMA operation), and the Block Attribute Register.
6. Begin operation for another function on the SDIO card.

#### 54.5.3.3.2 Resume

To resume the data transfer, a Resume command shall be issued:

1. To resume the suspended function, restore the context register with the saved value in step #5 of the Suspend operation above.
2. Send the Resume command. In the Transfer Type register, all bit fields are set to the value as if this were another ordinary data transfer, instead of a transfer resume (except the CMDTYP is set to 2'b10).
3. If the Resume command has responded, the data transfer will be resumed.

#### 54.5.3.4 ADMA Usage

To use the ADMA in a data transfer, the Host Driver must prepare the correct descriptor chain prior to sending the read/write command.

The steps to prepare the correct descriptor chain are:

1. Create a descriptor to set the data length that the current descriptor group is about to transfer. The data length should be even numbers of the block size.
2. Create another descriptor to transfer the data from the address setting in this descriptor. The data address must be at a page boundary (4kB address aligned).
3. If necessary, create a Link descriptor containing the address of the next descriptor. The descriptor group is created in steps 1 ~ 3.

4. Repeat steps 1 ~ 3 until all descriptors are created.
5. In the last descriptor, set the End flag to 1 and make sure the total length of all descriptors match the product of the block size and block number configured in the Block Attribute Register.
6. Set the ADMA System Address Register to the address of the first descriptor and set the DMAS field in the Protocol Control Register to 01 to select the ADMA.
7. Issue a write or read command with the DMAEN bit set to 1 in the Transfer Type Register.

Steps 1 ~ 5 are independent of step 6, so step 6 can finish before steps 1 ~ 5. Regarding the descriptor configuration, it is recommended not to use the Link descriptor as it requires extra system memory access.

### 54.5.3.5 Transfer Error

#### 54.5.3.5.1 CRC Error

It is possible at the end of a block transfer, that a write CRC status error or read CRC error occurs. For this type of error the latest block received shall be discarded. This is because the integrity of the data block is not guaranteed. It is recommended to discard the following data blocks and re-transfer the block from the corrupted one.

For a multi-block transfer, the Host Driver shall issue a CMD12 to abort the current process and start the transfer by a new data command. In this scenario, even when the AC12EN and BCEND bits are set, the uSDHC does not automatically send a CMD12 because the last block is not transferred. On the other hand, if it is within the last block that the CRC error occurs, an Auto CMD12 will be sent by the uSDHC. In this case, the Driver shall re-send or re-obtain the last block with a single block transfer.

#### 54.5.3.5.2 Internal DMA Error

During the data transfer with internal Simple DMA, if the DMA engine encounters some error on the AHB bus, the DMA operation is aborted and DMA Error interrupt is sent to the Host System. When acknowledged by such an interrupt, the Driver shall calculate the start address of data block in which the error occurs.

The start address can be calculated by either:

1. Reading the DMA System Address register. The error occurs during the previous burst. Taking the block size, the previous burst length and the start address of the next burst transfer into account, it is straight forward to obtain the start address of the corrupted block.

2. Reading the BLKCNT field of the Block Attribute register. By the number of blocks left, the total number to transfer, the start address of transfer, and the size of each block, the start address of corrupted block can be determined. When the BCEN bit is not set, the contents of the Block Attribute register does not change, so this method does not work.

When a DMA error occurs, it is recommended to abort the current transfer by means of a CMD12 (for multi block transfer), apply a reset for data, and re-start the transfer from the corrupted block to recover from the error.

#### 54.5.3.5.3 Transfer ADMA Error

There are 3 kinds of possible ADMA errors. The AHB transfer, invalid descriptor, and data-length mismatch errors. Whenever these errors occur, the DMA transfer stops and the corresponding error status bit is set.

For acknowledging the status, the Host Driver should recover the error as shown below and re-transfer from the place of interruption.

1. AHB transfer error: Such errors may occur during data transfer or descriptor fetch. For either scenario, it is recommended to retrieve the transfer context, reset for the data part and re-transfer the block that was corrupted, or the next block if no block is corrupted.
2. Invalid descriptor error: For such errors, it is recommended to retrieve the transfer context, reset for the data part and re-create the descriptor chain from the invalid descriptor and issue a new transfer. As the data to transfer now may be less than the previous setting, the data length configured in the new descriptor chain should match the new value.
3. Data-length mismatch error: It is similar to recover from this error. The Host Driver polls relating registers to retrieve the transfer context, apply a reset for the data part, configure a new descriptor chain, and make another transfer if there is data left. Like the previous scenario of the invalid descriptor error, the data length must match the new transfer.

#### 54.5.3.5.4 Auto CMD12 Error

After the last block of the multi block transfer is sent or received, and the AC12EN bit is set when the data transfer is initiated by the data command, the uSDHC automatically sends a CMD12 to the card to stop the transfer.

When errors with this command occur, it is recommended to the Driver to deal with the situations in the following manner:

1. Auto CMD12 response time-out. It is not certain whether the command is accepted by the card or not. The Driver should clear the Auto CMD12 error status bits and re-send the CMD12 until it is accepted by the card.
2. Auto CMD12 response CRC error. Since card responds to the CMD12, the card will abort the transfer. The Driver may ignore the error and clear the error status bit.
3. Auto CMD12 conflict error or not sent. The command is not sent, so the Driver shall send a CMD12 manually.

### 54.5.3.6 Card Interrupt

The external cards can inform the Host Controller by means of some special signals. For the SDIO card, it can be the low level on the DATA1 line during some special period. The uSDHC only monitors the DATA1 line and supports the SDIO interrupt.

When the SDIO interrupt is captured by the uSDHC, and the Host System is informed by the uSDHC asserting the uSDHC interrupt line, the interrupt service from the Host Driver is called.

As the interrupt source is controlled by the external card, the interrupt from the SDIO card must be serviced before the CINT bit is cleared by written 1. Refer to [Card Interrupt Handling](#) for the card interrupt handling flow.

### 54.5.4 Switch Function

SD/MMC cards can transfer data at bus widths other than 1-bit. Different speed mode are also defined. To enable these features, a "switch" command shall be issued by the Host Driver.

For SDIO cards, the high speed mode/DDR50/SDR50/SDR104 are enabled by writing the EHS bit in the CCCR register after the SHS bit is confirmed. For SD cards, the high speed mode/DDR50/SDR50/SDR104 are queried and enabled by a CMD6 (with the mnemonic symbol as SWITCH\_FUNC). For MMC cards, the high speed mode/HS200 are queried by a CMD8 and enabled by a CMD6 (with the mnemonic symbol as SWITCH).

The SDR4-bit, SDR8-bit, DDR4-bit and DDR8-bit width of the MMC is also enabled by the SWITCH command, but with a different argument.

These new functions can also be disabled by a software reset. For SDIO cards it can be done by setting the RES bit in the CCCR register. For other cards, it can be accomplished by issuing a CMD0. This method of restoring to the normal mode is not recommended because a complete identification process is needed before the card is ready for data transfer.

For the sake of simplicity, the following pseudocode examples do not show current capability check, which is recommended in the function switch process.

#### 54.5.4.1 Query, Enable and Disable SDIO High Speed Mode

```
enable_sdio_high_speed_mode(void)
{
    send CMD52 to query bit SHS at address 0x13;
    if (SHS bit is '0') report the SDIO card does not support high speed mode and return;
    send CMD52 to set bit EHS at address 0x13 and read after write to confirm EHS bit is set;
    change clock divisor value or configure the system clock feeding into uSDHC to generate the
    card_clk of around 50MHz;
    (data transactions like normal peers)
}
disable_sdio_high_speed_mode(void)
{
    send CMD52 to clear bit EHS at address 0x13 and read after write to confirm EHS bit is
    cleared;
    change clock divisor value or configure the system clock feeding into uSDHC to generate the
    card_clk of the desired value below 25MHz;
    (data transactions like normal peers)
}
```

#### 54.5.4.2 Query, Enable and Disable SD High Speed Mode/SDR50/SDR104/DDR50

```
enable_sd_speed_mode(void)
{
    set BLKCNT field to 1 (block), set BLKSIZE field to 64 (bytes);
    send CMD6, with argument 0xFFFFFx and read 64 bytes of data accompanying the R1 response;
    (high speed mode,x=1; SDR50,x=2; SDR104,x=3; DDR50,x=4;)
    wait data transfer done bit is set;
    check if the bit x of received 512 bits is set;
    if (bit 401 is '0') report the SD card does not support high speed mode and return;
    if (bit 402 is '0') report the SD card does not support SDR50 mode and return;
    if (bit 403 is '0') report the SD card does not support SDR104 mode and return;
    if (bit 404 is '0') report the SD card does not support DDR50 mode and return;
    send CMD6, with argument 0x80FFFFFx and read 64 bytes of data accompanying the R1 response;
    (high speed mode,x=1; SDR50,x=2; SDR104 x=3; DDR50 x=4;)
    check if the bit field 379~376 is 0xF;
    if (the bit field is 0xF) report the function switch failed and return;
    change clock divisor value or configure the system clock feeding into uSDHC to generate the
    card_clk of around 50MHz for high speed mode, 100Mhz for SDR50, 200Mhz for SDR104, 50Mhz for
    DDR50;
    (data transactions like normal peers)
}
disable_sd_speed_mode(void)
{
    set BLKCNT field to 1 (block), set BLKSIZE field to 64 (bytes);
    send CMD6, with argument 0x80FFFFF0 and read 64 bytes of data accompanying the R1 response;
    check if the bit field 379~376 is 0xF;
```



```

if (the bit field is 0xF) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of the desired value below 25MHz;
(data transactions like normal peers)
}

```

### 54.5.4.3 Query, Enable and Disable MMC High Speed Mode

```

enable_mmc_high_speed_mode(void)
{
send CMD9 to get CSD value of MMC;
check if the value of SPEC_VER field is 4 or above;
if (SPEC_VER value is less than 4) report the MMC does not support high speed mode and
return;
set BLKCNT field to 1 (block), set BLKSIZE field to 512 (bytes);
send CMD8 to get EXT_CSD value of MMC;
extract the value of CARD_TYPE field to check the 'high speed mode' in this MMC is 26MHz or
52MHz;
send CMD6 with argument 0x1B90100;
send CMD13 to wait card ready (busy line released);
send CMD8 to get EXT_CSD value of MMC;
check if HS_TIMING byte (byte number 185) is 1;
if (HS_TIMING is not 1) report MMC switching to high speed mode failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of around 26MHz or 52MHz according to the CARD_TYPE;
(data transactions like normal peers)
}
disable_mmc_high_speed_mode(void)
{
send CMD6 with argument 0x2B90100;
set BLKCNT field to 1 (block), set BLKSIZE field to 512 (bytes);
send CMD8 to get EXT_CSD value of MMC;
check if HS_TIMING byte (byte number 185) is 0;
if (HS_TIMING is not 0) report the function switch failed and return;
change clock divisor value or configure the system clock feeding into uSDHC to generate the
card_clk of the desired value below 20MHz;
(data transactions like normal peers)
}

```

### 54.5.4.4 Set MMC Bus Width

```

change_mmc_bus_width(void)
{
send CMD9 to get CSD value of MMC;
check if the value of SPEC_VER field is 4 or above;
if (SPEC_VER value is less than 4) report the MMC does not support multiple bit width and
return;
send CMD6 with argument 0x3B70x00; (8-bit(dual data rate), x=6; 4-bit(dual data rate), x=5; 8-
bit, x=2; 4-bit, x=1; 1-bit, x=0)
send CMD13 to wait card ready (busy line released);
(data transactions like normal peers)
}

```

## 54.5.5 ADMA Operation

### 54.5.5.1 ADMA1 Operation

```

Set_adma1_descriptor
{
if (to start data transfer) {
// Make sure the address is 4KB align.
Set 'Set' type descriptor;
{
Set Act bits to 01;
Set [31:12] bits data length (byte unit);
}
Set 'Tran' type descriptor;
{
Set Act bits to 10;
Set [31:12] bits address (4KB align);
}
}
else if (to fetch descriptor at non-continuous address) {
Set Act bits to 11;
Set [31:12] bits the next descriptor address (4KB aligned);
}
else { // other types of descriptor
Set Act bits accordingly
}
if (this descriptor is the last one) {
Set End bit to 1;
}
if (to generate interrupt for this descriptor) {
Set Int bit to 1;
}
Set Valid bit to 1;
}

```

### 54.5.5.2 ADMA2 Operation

```

Set_adma2_descriptor
{
if (to start data transfer) {
// Make sure the address is a 32-bit boundary (lower 2-bit are always '00').
Set higher 32-bit of descriptor for this data transfer initial address;
Set [31:16] bits data length (byte unit);
Set Act bits to '10';
}
else if (to fetch descriptor at non-continuous address) {
Set Act bits to '11';
// Make sure the address is 32-bit boundary (lower 2-bit are always set to '00').
Set higher 32-bit of descriptor for the next descriptor address;
}
else { // other types of descriptor
Set Act bits accordingly
}
if (this descriptor is the last one) {
Set 'End' bit '1';
}
if (to generate interrupt for this descriptor) {
Set 'Int' bit '1';
}
Set the 'Valid' bit to '1';
}

```

### 54.5.6 Fast Boot Operation

### 54.5.6.1 Normal fast boot flow

1. Software must configure `init_active` bit (system control register bit 27) to make sure 74 card clocks are finished.
2. Software must configure the MMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot), and bit 5 to 0 (normal fast boot), and bit 4 to select the ack mode or not. If the data will be sent through DMA mode, the software should configure bit 7 to enable the automatic stop at block gap feature, and configure bit 3-bit 0 to select the ack timeout value according to the SD CLK frequency.
3. Software then needs to configure the Block Attributes Register to set the block size and count. If in DDR fast boot mode, the block size only can be configured to 512 bytes.
4. Software must configure the Protocol control register to set DTW (data transfer width). If in DDR fast boot mode, DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software needs to configure the Command Argument Register to set argument if needed (no need in normal fast boot).
6. Software must configure the Transfer Type Register to start the boot process. In normal boot mode, `CMDINX`, `CMDTYP`, `RSPTYP`, `CICEN`, `CCEN`, `AC12EN`, `BCEN` and `DMAEN` are kept at the default value. `DPSEL` bit is set to 1, `DTDSEL` is set to 1, `MSBSEL` is set to 1.
7. `DMAEN` should be configured as 0 in polling mode. And if `BCEN` is configured as 1, it is recommended to configure the number of blocks in the Block Attributes Register to the maximum value. If in DDR fast boot mode, `DDR_EN` needs to be set to 1.
8. When the step 6 is configured, the boot process will begin. Software needs to poll the data buffer ready status to read the data from the buffer in time. If a boot timeout happens (ack times out or the first data read times out), an interrupt will be triggered, and software must configure MMC Boot Register to bit 6 to 0 to disable boot. This makes `CMD` high, then after at least 56 clocks, it is ready to begin a normal initialization process.
9. If there is no timeout, software needs to determine when the data read is finished and then configure MMC Boot Register bit 6 to 0 to disable boot. This will make `CMD` line high and command completed asserted. After at least 56 clocks, it is ready to begin normal initialization process.
10. Reset the host and then can begin the normal process.

### 54.5.6.2 Alternative fast boot flow

1. Software needs to configure init\_active bit (system control register bit 27) to make sure 74 card clocks are finished.
2. Software needs to configure MMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot), and bit 5 to 1 (alternative boot), and bit 4 to select the ack mode or not. If data needs to be sent through DMA mode, then configure bit 7 to enable the automatic stop at block gap feature. Software should also configure bit 3-bit 0 to select the ack timeout value according to the SD clock frequency.
3. Software then needs to configure Block Attributes Register to set the block size and count. If in DDR fast boot mode, the block size only can be configured to 512 bytes.
4. Software needs to configure the Protocol control register to set the DTW (data transfer width). If in ddr fast boot mode, DTW only can be configure to 4-bit/8-bit dataline mode.
5. Software needs to configure Command Argument Register to set argument to 0xFFFFFFFFFA.
6. Software needs to configure the Transfer Type Register to start the boot process by CMD0 with 0xFFFFFFFFFA argument . In alternative boot, CMDINX, CMDTYP, RSPTYP, C ICEN, CCCEN, AC12EN, BCEN and DMAEN are kept default value. DPSEL bit is set to 1, DTDSEL is set to 1, MSBSEL is set to 1. Note DMAEN should be configured as 0 in polling mode. And if BCEN is configured as 1 in polling mode, it is recommended to configure the block count in the Block Attributes Register to the maximum value. If in DDR fast boot mode, DDR\_EN needs to be set to 1.
7. When the step 6 is configured, the boot process will begin. Software needs to poll the data buffer ready status to read the data from the buffer in time. If there is a boot timeout (ack data timeout in 50ms or data timeout in 1s), the host will send out the interrupt and software needs to send CMD0 with reset and then configure the boot enable bit to 0 to stop this process..
8. If there is no time out, software needs to decide when to stop the boot process, and send out the CMD0 with reset and then after the command is completed, configure the MMC Boot Register bit 6 to stop the process. After 8 clocks from the command completion, the slave (card) is ready for the identification step.
9. Reset the host and then begin the normal process.

### 54.5.6.3 Fast boot application case (in DMA mode)

In the boot application case, because the image destination and the image size are contained in the beginning of the image, it is necessary to switch DMA parameters on the fly during MMC fast boot.

In fast boot, the host can use ADMA2 (Advanced DMA2) with two destinations.

The detail flow is described below:

1. Software needs to configure INIT\_ACTIVE bit (system control register bit 27) to make sure 74 card clocks are finished.
2. Software needs to configure the MMC Boot Register (offset 0xc4) bit 6 to 1 (enable boot); and bit 5 to 0 (normal fast boot) or 1 (alternative boot); and bit 4 to select the ack mode or not. In DMA mode, configure bit 7 to 1 to enable the automatic stop at block gap feature. Also configure bits[31-16] to set the (BLK\_CNT - VALUE1). Here VALUE1 is the value of the block count that needs to transfer the first time, so that that the host will stop at the block gap when the uSDHC controller gets VALUE1 blocks from the device. Also configure bits[3-0] to select the ack timeout value according to the SD clock frequency.
3. Software then needs to configure the Block Attributes Register to set block size and count. If in DDR fast boot mode, the block size only can be configured to 512 bytes. In DMA mode, it is recommended to set the block count (BLK\_CNT) to the max value (16'hffff).
4. Software needs to configure Protocol Control Register to set DTW (data transfer width). If in DDR fast boot mode, the DTW only can be configured to 4-bit/8-bit dataline mode.
5. Software enable ADMA2 by configuring Protocol Control Register bits [9-8].
6. Software need to set at least three pairs ADMA2 descriptor in boot memory (ie, in IRAM, at least 6 word). The first pair descriptor define the start address (ie, IRAM) and data length (ie, 512 byte \* VALUE1) of first part boot code. Software also need to set the second pair descriptor, the second start address (any value that is writeable), data length is suggest to set 1~2 word (record as VALUE2). Note: the second couple desc also transfer useful data even at least 1 word. Because our ADMA2 can't support 0 data\_length data transfer descriptor.
7. Software needs to configure Command Argument Register to set argument to 0xFFFFFFFF in alternative fast boot, and don't need set in normal fast boot.
8. Software needs to configure Transfer Type Register to start the boot process. CMDINX, CMDTYP, RSPTYP, CICEN, CCCEN, AC12EN, BCEN and DMAEN are kept default value. DPSEL bit is set to 1, DTDSEL is set to 1, MSBSEL is set to 1. DMAEN is configured as 1 in DMA mode. And if BCEN is configured as 1, better to configure blk no in Block Attributes Register to the max value. And if in ddr fast boot mode, DDR\_EN need to be set to 1.
9. When the step 8 is configured, boot process will begin, the first VALUE1 block number data has transfer. Software need to polling TC bit (bit1 in Interrupt Status Register) to determine first transfer is end. Also software need to polling BGE bit (bit2 in Interrupt Status Register) to determine if first transfer stop at block gap.
10. When TC, BGE bit is 1, . SW can analyzes the first code of VALUE1 block, initializes the new memory device, if required, and sets the third pair of descriptors to

define the start address and length of the remaining part of boot code(VALUE3 the remain boot code block). Remember set the last descriptor with END.

11. Software needs to configure MMC Boot Register (offset 0xc4) again. Set bit 6 to 1(enable boot); and bit 5 to 0(normal fast boot), to 1(alternative boot); and bit 4 to select the ack mode or not. In DMA mode, configure bit 7 to 1 for enable automatically stop at block gap feature. Also configure bit31-bit16 to set the  $(BLK\_CNT - (VALUE1+1+VALUE3))$ , that host will stop at block gap when *the uSDHC controller gets (VALUE1+1+VALUE3) blocks from device totally include the blocks received in step 9*. And need to configure bit 3-bit0 to select the ack timeout value according to the sd clk frequency. Please note, Software doesn't need to configure the *BLK\_CNT* again, because it's counted down automatically by the uSDHC controller.
12. Software needs to clear TC and BGE bit. And software needs to clear SABGREQ(bit 16 in Protocol control register), and set CREQ(bit17 Protocol control register) to 1 to resume the data transfer. Host will transfer the VALUE2 and VALUE3 data to the destination that is set by descriptor.
13. Software need to polling BGE bit to determine if the fast boot is over.

Note:

1. When ADMA boot flow is started, for uSDHC, it is like a normal ADMA read operation. So setting ADMA2 descriptor as the normal ADMA2 transfer.
2. Need a few words length memory to keep descriptor.
3. For the 1~2 word data in second descriptor setting, it is the useful data, so software need to deal the data due to the application case.

## 54.6 Commands for MMC/SD/SDIO

A table containing the list of commands for the MMC/SD/SDIO cards can be found here.

Refer to the corresponding specifications for more details about the command information.

There are four kinds of commands defined to control the MultiMediaCard:

1. broadcast commands (bc), no response.
2. broadcast commands with response (bcr), response from all cards simultaneously.
3. addressed (point-to-point) commands (ac), no data transfer on the DATA.
4. addressed (point-to-point) data transfer commands (adtc).

Response: a response is a token which is sent from the card to the host as an answer to a previously received command. A response is transferred serially on the CMD line.

**Table 54-7. Commands for MMC/SD/SDIO Cards**

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD0	bc	[31:0] stuff bits	-	GO_IDLE_STATE	Resets all MMC and SD memory cards to idle state.
CMD1	bcr	[31:0] OCR without busy	R3	SEND_OP_COND	Asks all MMC and SD Memory cards in idle state to send their operation conditions register contents in the response on the CMD line.
CMD2	bcr	[31:0] stuff bits	R2	ALL_SEND_CID	Asks all cards to send their CID numbers on the CMD line.
CMD3 <sup>1</sup>	ac	[31:6] RCA [15:0] stuff bits	R1 R6 (SDIO)	SET/ SEND_RELATIVE_ADDR	Assigns relative address to the card.
CMD4	bc	[31:0] DSR [15:0] stuff bits	-	SET_DSR	Programs the DSR of all cards.
CMD5	bc	[31:0] OCR without busy	R4	IO_SEND_OP_COND	Asks all SDIO cards in idle state to send their operation conditions register contents in the response on the CMD line.
CMD6 <sup>2</sup>	adtc	[31] Mode 0: Check function 1: Switch function [30:8] Reserved for function groups 6 ~ 3 (All 0 or 0xFFFF) [7:4] Function group1 for command system [3:0] Function group2 for access mode	R1	SWITCH_FUNC	Checks switch ability (mode 0) and switch card function (mode 1). Refer to "SD Physical Specification V1.1" for more details.
CMD6 <sup>3</sup>	ac	[31:26] Set to 0 [25:24] Access [23:16] Index [15:8] Value [7:3] Set to 0 [2:0] Cmd Set	R1b	SWITCH	Switches the mode of operation of the selected card or modifies the EXT_CSD registers. Refer to "The MultiMediaCard System Specification Version 4.0 Final draft 2" for more details.
CMD7	ac	[31:6] RCA [15:0] stuff bits	R1b	SELECT/ DESELECT_CARD	Toggles a card between the stand-by and transfer states or between the programming and disconnect states. In both cases, the card is selected by its own relative address and gets deselected by any other address. Address 0 deselects all.
CMD8	adtc	[31:0] stuff bits	R1	SEND_EXT_CSD	The card sends its EXT_CSD register as a block of data, with a block size of 512 bytes.

Table continues on the next page...

**Table 54-7. Commands for MMC/SD/SDIO Cards (continued)**

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD9	ac	[31:6] RCA [15:0] stuff bits	R2	SEND_CSD	Addressed card sends its card-specific data (CSD) on the CMD line.
CMD10	ac	[31:6] RCA [15:0] stuff bits	R2	SEND_CID	Addressed card sends its card-identification (CID) on the CMD line.
CMD11	adtc	[31:0] data address	R1	READ_DAT_UNTIL_STOP	Reads data stream from the card, starting at the given address, until a STOP_TRANSMISSION follows.
CMD12	ac	[31:0] stuff bits	R1b	STOP_TRANSMISSION	Forces the card to stop transmission.
CMD13	ac	[31:6] RCA [15:0] stuff bits	R1	SEND_STATUS	Addressed card sends its status register.
CMD14	Reserved				
CMD15	ac	[31:6] RCA [15:0] stuff bits	-	GO_INACTIVE_STATE	Sets the card to inactive state in order to protect the card stack against communication breakdowns.
CMD16	ac	[31:0] block length	R1	SET_BLOCKLEN	Sets the block length (in bytes) for all following block commands (read and write). Default block length is specified in the CSD.
CMD17	adtc	[31:0] data address	R1	READ_SINGLE_BLOCK	Reads a block of the size selected by the SET_BLOCKLEN command.
CMD18	adtc	[31:0] data address	R1	READ_MULTIPLE_BLOCK	Continuously transfers data blocks from card to host until interrupted by a stop command.
CMD19	Reserved				
CMD20	adtc	[31:0] data address	R1	WRITE_DAT_UNTIL_STOP	Writes data stream from the host, starting at the given address, until a STOP_TRANSMISSION follows.
CMD21-23	Reserved				
CMD24	adtc	[31:0] data address	R1	WRITE_BLOCK	Writes a block of the size selected by the SET_BLOCKLEN command.
CMD25	adtc	[31:0] data address	R1	WRITE_MULTIPLE_BLOCK	Continuously writes blocks of data until a STOP_TRANSMISSION follows.
CMD26	adtc	[31:0] stuff bits	R1	PROGRAM_CID	Programming of the card identification register. This command shall be issued only once per card. The card contains hardware to prevent this operation after the first programming. Normally this command is reserved for the manufacturer.
CMD27	adtc	[31:0] stuff bits	R1	PROGRAM_CSD	Programming of the programmable bits of the CSD.

*Table continues on the next page...*



**Table 54-7. Commands for MMC/SD/SDIO Cards (continued)**

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CMD28	ac	[31:0] data address	R1b	SET_WRITE_PROT	If the card has write protection features, this command sets the write protection bit of the addressed group. The properties of write protection are coded in the card specific data (WP_GRP_SIZE).
CMD29	ac	[31:0] data address	R1b	CLR_WRITE_PROT	If the card provides write protection features, this command clears the write protection bit of the addressed group.
CMD30	adtc	[31:0] write protect data address	R1	SEND_WRITE_PROT	If the card provides write protection features, this command asks the card to send the status of the write protection bits.
CMD31	Reserved				
CMD32	ac	[31:0] data address	R1	TAG_SECTOR_START	Sets the address of the first sector of the erase group.
CMD33	ac	[31:0] data address	R1	TAG_SECTOR_END	Sets the address of the last sector in a continuous range within the selection of a single sector to be selected for erase.
CMD34	ac	[31:0] data address	R1	UNTAG_SECTOR	Removes one previously selected sector from the erase selection.
CMD35	ac	[31:0] data address	R1	TAG_ERASE_GROUP_START	Sets the address of the first erase group within a range to be selected for erase.
CMD36	ac	[31:0] data address	R1	TAG_ERASE_GROUP_END	Sets the address of the last erase group within a continuous range to be selected for erase.
CMD37	ac	[31:0] data address	R1	UNTAG_ERASE_GROUP	Removes one previously selected erase group from the erase selection.
CMD38	ac	[31:0] stuff bits	R1b	ERASE	Erase all previously selected sectors.
CMD39	ac	[31:0] RCA [15] register write flag [14:8] register address [7:0] register data	R4	FAST_IO	Used to write and read 8-bit (register) data fields. The command addresses a card, and a register, and provides the data for writing if the write flag is set. The R4 response contains data read from the address register. This command accesses application dependent registers which are not defined in the MMC standard.
CMD40	bcr	[31:0] stuff bits	R5	GO_IRQ_STATE	Sets the system into interrupt mode.
CMD41	Reserved				

*Table continues on the next page...*

**Table 54-7. Commands for MMC/SD/SDIO Cards (continued)**

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
CDM42	adtc	[31:0] stuff bits	R1b	LOCK_UNLOCK	Used to set/reset the password or lock/unlock the card. The size of the data block is set by the SET_BLOCK_LEN command.
CMD43~51	Reserved				
CMD52	ac	[31:0] stuff bits	R5	IO_RW_DIRECT	Access a single register within the total 128k of register space in any I/O function.
CMD53	ac	[31:0] stuff bits	R5	IO_RW_EXTENDED	Accesses a multiple I/O register with a single command. Allows the reading or writing of a large number of I/O registers.
CMD54	Reserved				
CMD55	ac	[31:16] RCA [15:0] stuff bits	R1	APP_CMD	Indicates to the card that the next command is an application specific command rather than a standard command.
CMD56	adtc	[31:1] stuff bits [0]: RD/WR	R1b	GEN_CMD	Used either to transfer a data block to the card or to get a data block from the card for general purpose / application specific commands. The size of the data block is set by the SET_BLOCK_LEN command.
CMD57-59	Reserved				
CMD60	adtc	[31] WR [30:24] stuff bits [23:16] address [15:8] stuff bits [7:0] byte count	R1b	RW_MULTIPLE_REGISTER	These registers are used to control the behavior of the device and to retrieve status information regarding the operation of the device. All Status and Control registers are WORD (32-bit) in size and are WORD aligned. CMD60 shall be used to read and write these registers.
CMD61	adtc	[31] WR [30:16] stuff bits [15:0] data unit count	R1b	RW_MULTIPLE_BLOCK	The host issues a RW_MULTIPLE_BLOCK (CMD61) to begin the data transfer.
CMD62-63	Reserved				
ACMD6 <sup>4</sup>	ac	[31:2] stuff bits [1:0] bus width	R1	SET_BUS_WIDTH	Defines the data bus width ('00'=1bit or '10'=4bit bus) to be used for data transfer. The allowed data bus widths are given in SCR register.
ACMD13 <sup>4</sup>	adtc	[31:0] stuff bits	R1	SD_STATUS	Send the SD Memory Card status.
ACMD22 <sup>4</sup>	adtc	[31:0] stuff bits	R1	SEND_NUM_WR_SECTORS	Send the number of the written sectors (without errors). Responds with 32-bit plus the CRC data block.

Table continues on the next page...

**Table 54-7. Commands for MMC/SD/SDIO Cards (continued)**

CMD INDEX	Type	Argument	Response type	Abbreviation	Description
ACMD23 <sup>4</sup>	ac	[31:23] stuff bits [22:0] Number of blocks	R1	SET_WR_BLK_ERASE_COUNT	Set the number of write blocks to be pre-erased before writing (to be used for fast Multiple Block WR command). "1"=default(one write block).
ACMD41 <sup>4</sup>	bcr	[31:0] OCR	R3	SD_APP_OP_COND	Asks the accessed card to send its operating condition register (OCR) contents in the response on the CMD line.
ACMD42 <sup>4</sup>	ac	[31:1] stuff bits [0] set_cd	R1	SET_CLR_CARD_DETECT	Connect(1)/Disconnect(0) the 50KOhm pull-up resistor on CD_B/ DATA3 of the card.
ACMD51 <sup>4</sup>	adtc	[31:0] stuff bits	R1	SEND_SCR	Reads the SD Configuration Register (SCR).

1. CMD3 differs for MMC and SD cards. For MMC cards, it is referred to as SET\_RELATIVE\_ADDR, with a response type of R1. For SD cards, it is referred to as SEND\_RELATIVE\_ADDR, with a response type of R6 (with RCA inside).
2. CMD6 differs completely between high speed MMC cards and high speed SD cards. Command SWITCH\_FUNC is for high speed SD cards.
3. Command SWITCH is for high speed MMC cards. The Index field can contain any value from 0-255, but only values 0-191 are valid. If the Index value is in the 192-255 range the card does not perform any modification and the SWITCH\_ERROR status bit in the EXT\_CSD register is set. The Access Bits are shown in [Table 2](#).
4. ACMDs shall be preceded with the APP\_CMD command. (Commands listed are used for SD only, other SD commands not listed are not supported on this module).

The Access Bits for the EXT\_CSD Access Modes are shown below.

**Table 54-8. EXT\_CSD Access Modes**

Bits	Access Name	Operation
00	Command Set	The command set is changed according to the Cmd Set field of the argument
01	Set Bits	The bits in the pointed byte are set, according to the 1 bits in the Value field.
10	Clear Bits	The bits in the pointed byte are cleared, according to the 1 bits in the Value field.
11	Write Byte	The Value field is written into the pointed byte.

## 54.7 Software Restrictions

### 54.7.1 Initialization Active

The driver cannot set INITA bit in System Control register when any of the command line or data lines is active, so the driver must ensure both CDIHB and CIHB bits are cleared.

### 54.7.2 Software Polling Procedure

For polling read or write, once the software begins a buffer read or write, it must access exactly the number of times as the values set in the Watermark Level Register; moreover, if the block size is not a multiple of the value in Watermark Level Register (read and write respectively), the software must access exactly the remaining number of words at the end of each block.

For example, for a read operation, if the RD\_WML is 4, indicating the watermark level is 16 bytes, block size is 40 bytes, and the block number is 2, then the access times for the burst sequence in the whole transfer process must be 4, 4, 2, 4, 4, 2.

### 54.7.3 Suspend Operation

In order to suspend the data transfer, the software must inform uSDHC that the suspend command is successfully accepted. To achieve this, after the Suspend command is accepted by the SDIO card, software must send another normal command marked as suspend command (CMDTYP bits set as '01') to inform uSDHC that the transfer is suspended.

If software needs to resume the suspended transfer, it should read the value in BLKCNT register to save the remaining number of blocks before sending the normal command marked as suspend, otherwise on sending such 'suspend' command, uSDHC will regard the current transfer is aborted and change BLKCNT register to its original value, instead of keeping the remained number of blocks.

### 54.7.4 Data Length Setting

For either ADMA (ADMA1 or ADMA2) transfer, the data in the data buffer must be word aligned, so the data length set in the descriptor must be a multiple of 4.

### 54.7.5 (A)DMA Address Setting

To configure ADMA1/ADMA2/DMA address register, when TC bit is set, the register will always update itself with the internal address value to support dynamic address synchronization, so the software must ensure that the TC bit is cleared prior to configuring ADMA1/ADMA2/DMA address register.

### 54.7.6 Data Port Access

Data Port does not support parallel access. For example, during an external DMA access, it is not allowed to write any data to the Data Port by CPU; or during a CPU read operation, it is also prohibited to write any data to the Data Port, by either CPU or external DMA. Otherwise the data would be corrupted inside the uSDHC buffer.

### 54.7.7 Change Clock Frequency

uSDHC does not automatically gate off the card clock when the Host Driver changes the clock frequency. To prevent possible glitch on the card clock, clear the FRC\_SDCLK\_ON bit when changing clock divisor value(SDCLKFS or DVS in System Control Register) or setting RSTA bit.

Also before changing the clock divisor value, Host Driver should make sure the SDSTB bit is high.

### 54.7.8 Multi-block Read

For pre-defined multi-block read operation, i.e., the number of blocks to read has been defined by previous CMD23 for MMC, or pre-defined number of blocks in CMD53 for SDIO/SDCombo, or whatever multi-block read without abort command at card side, an abort command, either automatic or manual CMD12/CMD52, is still required by uSDHC after the pre-defined number of blocks are done, to drive the internal state machine to idle mode.

In this case, the card may not respond to this extra abort command and uSDHC will get Response Timeout. It is recommended to manually send an abort command with RSPTYP[1:0] both bits cleared.

## 54.8 uSDHC Memory Map/Register Definition

This section includes the module memory map and detailed descriptions of all registers.

See the table below for the register memory map for the uSDHC. All these registers only support 32-bit accesses.

### NOTE

The uSDHC registers are 32-bit wide and only support 32-bit access.

### uSDHC memory map

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
219_0000	DMA System Address (uSDHC1_DS_ADDR)	32	R/W	0000_0000h	<a href="#">54.8.1/3383</a>
219_0004	Block Attributes (uSDHC1_BLK_ATT)	32	R/W	0000_0000h	<a href="#">54.8.2/3384</a>
219_0008	Command Argument (uSDHC1_CMD_ARG)	32	R/W	0000_0000h	<a href="#">54.8.3/3385</a>
219_000C	Command Transfer Type (uSDHC1_CMD_XFR_TYP)	32	R/W	0000_0000h	<a href="#">54.8.4/3385</a>
219_0010	Command Response0 (uSDHC1_CMD_RSP0)	32	R	0000_0000h	<a href="#">54.8.5/3389</a>
219_0014	Command Response1 (uSDHC1_CMD_RSP1)	32	R	0000_0000h	<a href="#">54.8.6/3389</a>
219_0018	Command Response2 (uSDHC1_CMD_RSP2)	32	R	0000_0000h	<a href="#">54.8.7/3390</a>
219_001C	Command Response3 (uSDHC1_CMD_RSP3)	32	R	0000_0000h	<a href="#">54.8.8/3390</a>
219_0020	Data Buffer Access Port (uSDHC1_DATA_BUFF_ACC_PORT)	32	R/W	0000_0000h	<a href="#">54.8.9/3392</a>
219_0024	Present State (uSDHC1_PRES_STATE)	32	R	0000_8080h	<a href="#">54.8.10/3392</a>
219_0028	Protocol Control (uSDHC1_PROT_CTRL)	32	R/W	0880_0020h	<a href="#">54.8.11/3398</a>
219_002C	System Control (uSDHC1_SYS_CTRL)	32	R/W	8080_800Fh	<a href="#">54.8.12/3403</a>
219_0030	Interrupt Status (uSDHC1_INT_STATUS)	32	w1c	0000_0000h	<a href="#">54.8.13/3406</a>
219_0034	Interrupt Status Enable (uSDHC1_INT_STATUS_EN)	32	R/W	157F_413Fh	<a href="#">54.8.14/3412</a>
219_0038	Interrupt Signal Enable (uSDHC1_INT_SIGNAL_EN)	32	R/W	0000_0000h	<a href="#">54.8.15/3415</a>
219_003C	Auto CMD12 Error Status (uSDHC1_AUTOCMD12_ERR_STATUS)	32	R	0000_0000h	<a href="#">54.8.16/3418</a>
219_0040	Host Controller Capabilities (uSDHC1_HOST_CTRL_CAP)	32	R	07F3_B407h	<a href="#">54.8.17/3421</a>
219_0044	Watermark Level (uSDHC1_WTMK_LVL)	32	R/W	0810_0810h	<a href="#">54.8.18/3424</a>
219_0048	Mixer Control (uSDHC1_MIX_CTRL)	32	R/W	8000_0000h	<a href="#">54.8.19/3425</a>

*Table continues on the next page...*

## uSDHC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
219_0050	Force Event (uSDHC1_FORCE_EVENT)	32	W (always reads 0)	0000_0000h	<a href="#">54.8.20/3427</a>
219_0054	ADMA Error Status Register (uSDHC1_ADMA_ERR_STATUS)	32	R	0000_0000h	<a href="#">54.8.21/3430</a>
219_0058	ADMA System Address (uSDHC1_ADMA_SYS_ADDR)	32	R/W	0000_0000h	<a href="#">54.8.22/3432</a>
219_0060	DLL (Delay Line) Control (uSDHC1_DLL_CTRL)	32	R/W	0000_0000h	<a href="#">54.8.23/3433</a>
219_0064	DLL Status (uSDHC1_DLL_STATUS)	32	R	0000_0000h	<a href="#">54.8.24/3435</a>
219_0068	CLK Tuning Control and Status (uSDHC1_CLK_TUNE_CTRL_STATUS)	32	R/W	0000_0000h	<a href="#">54.8.25/3436</a>
219_00C0	Vendor Specific Register (uSDHC1_VEND_SPEC)	32	R/W	2000_7809h	<a href="#">54.8.26/3438</a>
219_00C4	MMC Boot Register (uSDHC1_MMC_BOOT)	32	R/W	0000_0000h	<a href="#">54.8.27/3441</a>
219_00C8	Vendor Specific 2 Register (uSDHC1_VEND_SPEC2)	32	R/W	0000_0006h	<a href="#">54.8.28/3442</a>
219_00CC	Tuning Control Register (uSDHC1_TUNING_CTRL)	32	R/W	0021_2800h	<a href="#">54.8.29/3444</a>
219_4000	DMA System Address (uSDHC2_DS_ADDR)	32	R/W	0000_0000h	<a href="#">54.8.1/3383</a>
219_4004	Block Attributes (uSDHC2_BLK_ATT)	32	R/W	0000_0000h	<a href="#">54.8.2/3384</a>
219_4008	Command Argument (uSDHC2_CMD_ARG)	32	R/W	0000_0000h	<a href="#">54.8.3/3385</a>
219_400C	Command Transfer Type (uSDHC2_CMD_XFR_TYP)	32	R/W	0000_0000h	<a href="#">54.8.4/3385</a>
219_4010	Command Response0 (uSDHC2_CMD_RSP0)	32	R	0000_0000h	<a href="#">54.8.5/3389</a>
219_4014	Command Response1 (uSDHC2_CMD_RSP1)	32	R	0000_0000h	<a href="#">54.8.6/3389</a>
219_4018	Command Response2 (uSDHC2_CMD_RSP2)	32	R	0000_0000h	<a href="#">54.8.7/3390</a>
219_401C	Command Response3 (uSDHC2_CMD_RSP3)	32	R	0000_0000h	<a href="#">54.8.8/3390</a>
219_4020	Data Buffer Access Port (uSDHC2_DATA_BUFF_ACC_PORT)	32	R/W	0000_0000h	<a href="#">54.8.9/3392</a>
219_4024	Present State (uSDHC2_PRES_STATE)	32	R	0000_8080h	<a href="#">54.8.10/3392</a>
219_4028	Protocol Control (uSDHC2_PROT_CTRL)	32	R/W	0880_0020h	<a href="#">54.8.11/3398</a>
219_402C	System Control (uSDHC2_SYS_CTRL)	32	R/W	8080_800Fh	<a href="#">54.8.12/3403</a>
219_4030	Interrupt Status (uSDHC2_INT_STATUS)	32	w1c	0000_0000h	<a href="#">54.8.13/3406</a>
219_4034	Interrupt Status Enable (uSDHC2_INT_STATUS_EN)	32	R/W	157F_413Fh	<a href="#">54.8.14/3412</a>
219_4038	Interrupt Signal Enable (uSDHC2_INT_SIGNAL_EN)	32	R/W	0000_0000h	<a href="#">54.8.15/3415</a>

Table continues on the next page...

### uSDHC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
219_403C	Auto CMD12 Error Status (uSDHC2_AUTOCMD12_ERR_STATUS)	32	R	0000_0000h	<a href="#">54.8.16/3418</a>
219_4040	Host Controller Capabilities (uSDHC2_HOST_CTRL_CAP)	32	R	07F3_B407h	<a href="#">54.8.17/3421</a>
219_4044	Watermark Level (uSDHC2_WTMK_LVL)	32	R/W	0810_0810h	<a href="#">54.8.18/3424</a>
219_4048	Mixer Control (uSDHC2_MIX_CTRL)	32	R/W	8000_0000h	<a href="#">54.8.19/3425</a>
219_4050	Force Event (uSDHC2_FORCE_EVENT)	32	W (always reads 0)	0000_0000h	<a href="#">54.8.20/3427</a>
219_4054	ADMA Error Status Register (uSDHC2_ADMA_ERR_STATUS)	32	R	0000_0000h	<a href="#">54.8.21/3430</a>
219_4058	ADMA System Address (uSDHC2_ADMA_SYS_ADDR)	32	R/W	0000_0000h	<a href="#">54.8.22/3432</a>
219_4060	DLL (Delay Line) Control (uSDHC2_DLL_CTRL)	32	R/W	0000_0000h	<a href="#">54.8.23/3433</a>
219_4064	DLL Status (uSDHC2_DLL_STATUS)	32	R	0000_0000h	<a href="#">54.8.24/3435</a>
219_4068	CLK Tuning Control and Status (uSDHC2_CLK_TUNE_CTRL_STATUS)	32	R/W	0000_0000h	<a href="#">54.8.25/3436</a>
219_40C0	Vendor Specific Register (uSDHC2_VEND_SPEC)	32	R/W	2000_7809h	<a href="#">54.8.26/3438</a>
219_40C4	MMC Boot Register (uSDHC2_MMC_BOOT)	32	R/W	0000_0000h	<a href="#">54.8.27/3441</a>
219_40C8	Vendor Specific 2 Register (uSDHC2_VEND_SPEC2)	32	R/W	0000_0006h	<a href="#">54.8.28/3442</a>
219_40CC	Tuning Control Register (uSDHC2_TUNING_CTRL)	32	R/W	0021_2800h	<a href="#">54.8.29/3444</a>
219_8000	DMA System Address (uSDHC3_DS_ADDR)	32	R/W	0000_0000h	<a href="#">54.8.1/3383</a>
219_8004	Block Attributes (uSDHC3_BLK_ATT)	32	R/W	0000_0000h	<a href="#">54.8.2/3384</a>
219_8008	Command Argument (uSDHC3_CMD_ARG)	32	R/W	0000_0000h	<a href="#">54.8.3/3385</a>
219_800C	Command Transfer Type (uSDHC3_CMD_XFR_TYP)	32	R/W	0000_0000h	<a href="#">54.8.4/3385</a>
219_8010	Command Response0 (uSDHC3_CMD_RSP0)	32	R	0000_0000h	<a href="#">54.8.5/3389</a>
219_8014	Command Response1 (uSDHC3_CMD_RSP1)	32	R	0000_0000h	<a href="#">54.8.6/3389</a>
219_8018	Command Response2 (uSDHC3_CMD_RSP2)	32	R	0000_0000h	<a href="#">54.8.7/3390</a>
219_801C	Command Response3 (uSDHC3_CMD_RSP3)	32	R	0000_0000h	<a href="#">54.8.8/3390</a>
219_8020	Data Buffer Access Port (uSDHC3_DATA_BUFF_ACC_PORT)	32	R/W	0000_0000h	<a href="#">54.8.9/3392</a>
219_8024	Present State (uSDHC3_PRES_STATE)	32	R	0000_8080h	<a href="#">54.8.10/3392</a>
219_8028	Protocol Control (uSDHC3_PROT_CTRL)	32	R/W	0880_0020h	<a href="#">54.8.11/3398</a>

Table continues on the next page...



## uSDHC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
219_802C	System Control (uSDHC3_SYS_CTRL)	32	R/W	8080_800Fh	<a href="#">54.8.12/3403</a>
219_8030	Interrupt Status (uSDHC3_INT_STATUS)	32	w1c	0000_0000h	<a href="#">54.8.13/3406</a>
219_8034	Interrupt Status Enable (uSDHC3_INT_STATUS_EN)	32	R/W	157F_413Fh	<a href="#">54.8.14/3412</a>
219_8038	Interrupt Signal Enable (uSDHC3_INT_SIGNAL_EN)	32	R/W	0000_0000h	<a href="#">54.8.15/3415</a>
219_803C	Auto CMD12 Error Status (uSDHC3_AUTOCMD12_ERR_STATUS)	32	R	0000_0000h	<a href="#">54.8.16/3418</a>
219_8040	Host Controller Capabilities (uSDHC3_HOST_CTRL_CAP)	32	R	07F3_B407h	<a href="#">54.8.17/3421</a>
219_8044	Watermark Level (uSDHC3_WTMK_LVL)	32	R/W	0810_0810h	<a href="#">54.8.18/3424</a>
219_8048	Mixer Control (uSDHC3_MIX_CTRL)	32	R/W	8000_0000h	<a href="#">54.8.19/3425</a>
219_8050	Force Event (uSDHC3_FORCE_EVENT)	32	W (always reads 0)	0000_0000h	<a href="#">54.8.20/3427</a>
219_8054	ADMA Error Status Register (uSDHC3_ADMA_ERR_STATUS)	32	R	0000_0000h	<a href="#">54.8.21/3430</a>
219_8058	ADMA System Address (uSDHC3_ADMA_SYS_ADDR)	32	R/W	0000_0000h	<a href="#">54.8.22/3432</a>
219_8060	DLL (Delay Line) Control (uSDHC3_DLL_CTRL)	32	R/W	0000_0000h	<a href="#">54.8.23/3433</a>
219_8064	DLL Status (uSDHC3_DLL_STATUS)	32	R	0000_0000h	<a href="#">54.8.24/3435</a>
219_8068	CLK Tuning Control and Status (uSDHC3_CLK_TUNE_CTRL_STATUS)	32	R/W	0000_0000h	<a href="#">54.8.25/3436</a>
219_80C0	Vendor Specific Register (uSDHC3_VEND_SPEC)	32	R/W	2000_7809h	<a href="#">54.8.26/3438</a>
219_80C4	MMC Boot Register (uSDHC3_MMC_BOOT)	32	R/W	0000_0000h	<a href="#">54.8.27/3441</a>
219_80C8	Vendor Specific 2 Register (uSDHC3_VEND_SPEC2)	32	R/W	0000_0006h	<a href="#">54.8.28/3442</a>
219_80CC	Tuning Control Register (uSDHC3_TUNING_CTRL)	32	R/W	0021_2800h	<a href="#">54.8.29/3444</a>
219_C000	DMA System Address (uSDHC4_DS_ADDR)	32	R/W	0000_0000h	<a href="#">54.8.1/3383</a>
219_C004	Block Attributes (uSDHC4_BLK_ATT)	32	R/W	0000_0000h	<a href="#">54.8.2/3384</a>
219_C008	Command Argument (uSDHC4_CMD_ARG)	32	R/W	0000_0000h	<a href="#">54.8.3/3385</a>
219_C00C	Command Transfer Type (uSDHC4_CMD_XFR_TYP)	32	R/W	0000_0000h	<a href="#">54.8.4/3385</a>
219_C010	Command Response0 (uSDHC4_CMD_RSP0)	32	R	0000_0000h	<a href="#">54.8.5/3389</a>
219_C014	Command Response1 (uSDHC4_CMD_RSP1)	32	R	0000_0000h	<a href="#">54.8.6/3389</a>

Table continues on the next page...

### uSDHC memory map (continued)

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
219_C018	Command Response2 (uSDHC4_CMD_RSP2)	32	R	0000_0000h	<a href="#">54.8.7/3390</a>
219_C01C	Command Response3 (uSDHC4_CMD_RSP3)	32	R	0000_0000h	<a href="#">54.8.8/3390</a>
219_C020	Data Buffer Access Port (uSDHC4_DATA_BUFF_ACC_PORT)	32	R/W	0000_0000h	<a href="#">54.8.9/3392</a>
219_C024	Present State (uSDHC4_PRES_STATE)	32	R	0000_8080h	<a href="#">54.8.10/3392</a>
219_C028	Protocol Control (uSDHC4_PROT_CTRL)	32	R/W	0880_0020h	<a href="#">54.8.11/3398</a>
219_C02C	System Control (uSDHC4_SYS_CTRL)	32	R/W	8080_800Fh	<a href="#">54.8.12/3403</a>
219_C030	Interrupt Status (uSDHC4_INT_STATUS)	32	w1c	0000_0000h	<a href="#">54.8.13/3406</a>
219_C034	Interrupt Status Enable (uSDHC4_INT_STATUS_EN)	32	R/W	157F_413Fh	<a href="#">54.8.14/3412</a>
219_C038	Interrupt Signal Enable (uSDHC4_INT_SIGNAL_EN)	32	R/W	0000_0000h	<a href="#">54.8.15/3415</a>
219_C03C	Auto CMD12 Error Status (uSDHC4_AUTOCMD12_ERR_STATUS)	32	R	0000_0000h	<a href="#">54.8.16/3418</a>
219_C040	Host Controller Capabilities (uSDHC4_HOST_CTRL_CAP)	32	R	07F3_B407h	<a href="#">54.8.17/3421</a>
219_C044	Watermark Level (uSDHC4_WTMK_LVL)	32	R/W	0810_0810h	<a href="#">54.8.18/3424</a>
219_C048	Mixer Control (uSDHC4_MIX_CTRL)	32	R/W	8000_0000h	<a href="#">54.8.19/3425</a>
219_C050	Force Event (uSDHC4_FORCE_EVENT)	32	W (always reads 0)	0000_0000h	<a href="#">54.8.20/3427</a>
219_C054	ADMA Error Status Register (uSDHC4_ADMA_ERR_STATUS)	32	R	0000_0000h	<a href="#">54.8.21/3430</a>
219_C058	ADMA System Address (uSDHC4_ADMA_SYS_ADDR)	32	R/W	0000_0000h	<a href="#">54.8.22/3432</a>
219_C060	DLL (Delay Line) Control (uSDHC4_DLL_CTRL)	32	R/W	0000_0000h	<a href="#">54.8.23/3433</a>
219_C064	DLL Status (uSDHC4_DLL_STATUS)	32	R	0000_0000h	<a href="#">54.8.24/3435</a>
219_C068	CLK Tuning Control and Status (uSDHC4_CLK_TUNE_CTRL_STATUS)	32	R/W	0000_0000h	<a href="#">54.8.25/3436</a>
219_C0C0	Vendor Specific Register (uSDHC4_VEND_SPEC)	32	R/W	2000_7809h	<a href="#">54.8.26/3438</a>
219_C0C4	MMC Boot Register (uSDHC4_MMC_BOOT)	32	R/W	0000_0000h	<a href="#">54.8.27/3441</a>
219_C0C8	Vendor Specific 2 Register (uSDHC4_VEND_SPEC2)	32	R/W	0000_0006h	<a href="#">54.8.28/3442</a>

Table continues on the next page...

**uSDHC memory map (continued)**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
219_C0CC	Tuning Control Register (uSDHC4_TUNING_CTRL)	32	R/W	0021_2800h	<a href="#">54.8.29/3444</a>

**54.8.2 DMA System Address (uSDHCx\_DS\_ADDR)**

This register contains the physical system memory address used for DMA transfers.

Address: Base address + 0h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DS_ADDR[31:2]																														0	
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**uSDHCx\_DS\_ADDR field descriptions**

Field	Description
31–2 DS_ADDR[31:2]	<p><b>DMA System Address:</b></p> <p>This register contains the 32-bit system memory address for a DMA transfer. Since the address must be word (4 bytes) aligned, the least 2 bits are reserved, always 0. When the uSDHC stops a DMA transfer, this register points to the system address of the next contiguous data position. It can be accessed only when no transaction is executing (i.e. after a transaction has stopped). Read operation during transfers may return an invalid value. The Host Driver shall initialize this register before starting a DMA transaction. After DMA has stopped, the system address of the next contiguous data position can be read from this register.</p> <p>This register is protected during a data transfer. When data lines are active, write to this register is ignored. The Host driver shall wait, until the DLA bit in the Present State register is cleared, before writing to this register.</p> <p>The uSDHC internal DMA does not support a virtual memory system. It only supports continuous physical memory access. And due to AHB burst limitations, if the burst must cross the 1 KB boundary, uSDHC will automatically change SEQ burst type to NSEQ.</p> <p>Since this register supports dynamic address reflecting, when TC bit is set, it automatically alters the value of internal address counter, so SW cannot change this register when TC bit is set. Such restriction is also listed in <a href="#">Software Restrictions</a>.</p>
1–0 Reserved	This read-only field is reserved and always has the value 0.

### 54.8.3 Block Attributes (uSDHCx\_BLK\_ATT)

This register is used to configure the number of data blocks and the number of bytes in each block.

Address: Base address + 4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																		0														
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

#### uSDHCx\_BLK\_ATT field descriptions

Field	Description
31–16 BLKCNT	<p>Blocks Count For Current Transfer:</p> <p>This register is enabled when the Block Count Enable bit in the Transfer Mode register is set to 1 and is valid only for multiple block transfers. For single block transfer, this register will always read as 1. The Host Driver shall set this register to a value between 1 and the maximum block count. The uSDHC decrements the block count after each block transfer and stops when the count reaches zero. Setting the block count to 0 results in no data blocks being transferred.</p> <p>This register should be accessed only when no transaction is executing (i.e. after transactions are stopped). During data transfer, read operations on this register may return an invalid value and write operations are ignored.</p> <p>When saving transfer content as a result of a Suspend command, the number of blocks yet to be transferred can be determined by reading this register. The reading of this register should be applied after transfer is paused by stop at block gap operation and before sending the command marked as suspend. This is because when Suspend command is sent out, uSDHC will regard the current transfer is aborted and change BLKCNT register back to its original value instead of keeping the dynamical indicator of remained block count.</p> <p>When restoring transfer content prior to issuing a Resume command, the Host Driver shall restore the previously saved block count.</p> <p><b>NOTE:</b> Although the BLKCNT field is 0 after reset, the read of reset value is 0x1. This is because when MSBSEL bit is indicating a single block transfer, the read value of BLKCNT is always 1.</p> <p>FFFF 65535 blocks 0002 2 blocks 0001 1 block 0000 Stop Count</p>
15–13 Reserved	This read-only field is reserved and always has the value 0.
12–0 BLKSIZE[12:0]	<p>Transfer Block Size:</p> <p>This register specifies the block size for block data transfers. Values ranging from 1 byte up to the maximum buffer size can be set. It can be accessed only when no transaction is executing (i.e. after a transaction has stopped). Read operations during transfers may return an invalid value, and write operations will be ignored.</p> <p>1000 4096 Bytes</p>

Table continues on the next page...

**uSDHCx\_BLK\_ATT field descriptions (continued)**

Field	Description
800	2048 Bytes
200	512 Bytes
1FF	511 Bytes
004	4 Bytes
003	3 Bytes
002	2 Bytes
001	1 Byte
000	No data transfer

**54.8.4 Command Argument (uSDHCx\_CMD\_ARG)**

This register contains the SD/MMC Command Argument.

Address: Base address + 8h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDARG[31:0]																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

**uSDHCx\_CMD\_ARG field descriptions**

Field	Description
31–0 CMDARG[31:0]	Command Argument: The SD/MMC Command Argument is specified as bits 39-8 of the Command Format in the SD or MMC Specification. This register is write protected when the Command Inhibit (CMD) bit in the Present State register is set.

**54.8.5 Command Transfer Type (uSDHCx\_CMD\_XFR\_TYP)**

This register is used to control the operation of data transfers. The Host Driver shall set this register before issuing a command followed by a data transfer, or before issuing a Resume command. To prevent data loss, the uSDHC prevents writing to the bits, that are involved in the data transfer of this register, when data transfer is active. These bits are DPSEL, MBSEL, DTDSEL, AC12EN, BCEN and DMAEN.

The Host Driver shall check the Command Inhibit DAT bit (CDIHB) and the Command Inhibit CMD bit (CIHB) in the Present State register before writing to this register. When the CDIHB bit in the Present State register is set, any attempt to send a command with data by writing to this register is ignored; when the CIHB bit is set, any write to this register is ignored.

On sending commands with data transfer involved, it is mandatory that the block size is non-zero. Block count must also be non-zero, or indicated as single block transfer (bit 5 of this register is '0' when written), or block count is disabled (bit 1 of this register is '0' when written), otherwise uSDHC will ignore the sending of this command and do nothing. For write command, with all above restrictions, it is also mandatory that the write protect switch is not active (WPSPL bit of Present State Register is '1'), otherwise uSDHC will also ignore the command.

If the commands with data transfer does not receive the response in 64 clock cycles, i.e., response time-out, uSDHC will regard the external device does not accept the command and abort the data transfer. In this scenario, the driver should issue the command again to re-try the transfer. It is also possible that for some reason the card responds the command but uSDHC does not receive the response, and if it is internal DMA (either simple DMA or ADMA) read operation, the external system memory is over-written by the internal DMA with data sent back from the card.

The table below shows the summary of how register settings determine the type of data transfer.

**Table 54-53. Transfer Type Register Setting for Various Transfer Types**

Multi/Single Block Select	Block Count Enable	Block Count	Function
0	Don't Care	Don't Care	Single Transfer
1	0	Don't Care	Infinite Transfer
1	1	Positive Number	Multiple Transfer
1	1	Zero	No Data Transfer

The table below shows the relationship between the Command Index Check Enable and the Command CRC Check Enable, in regards to the Response Type bits as well as the name of the response type.

**Table 54-54. Relationship Between Parameters and the Name of the Response Type**

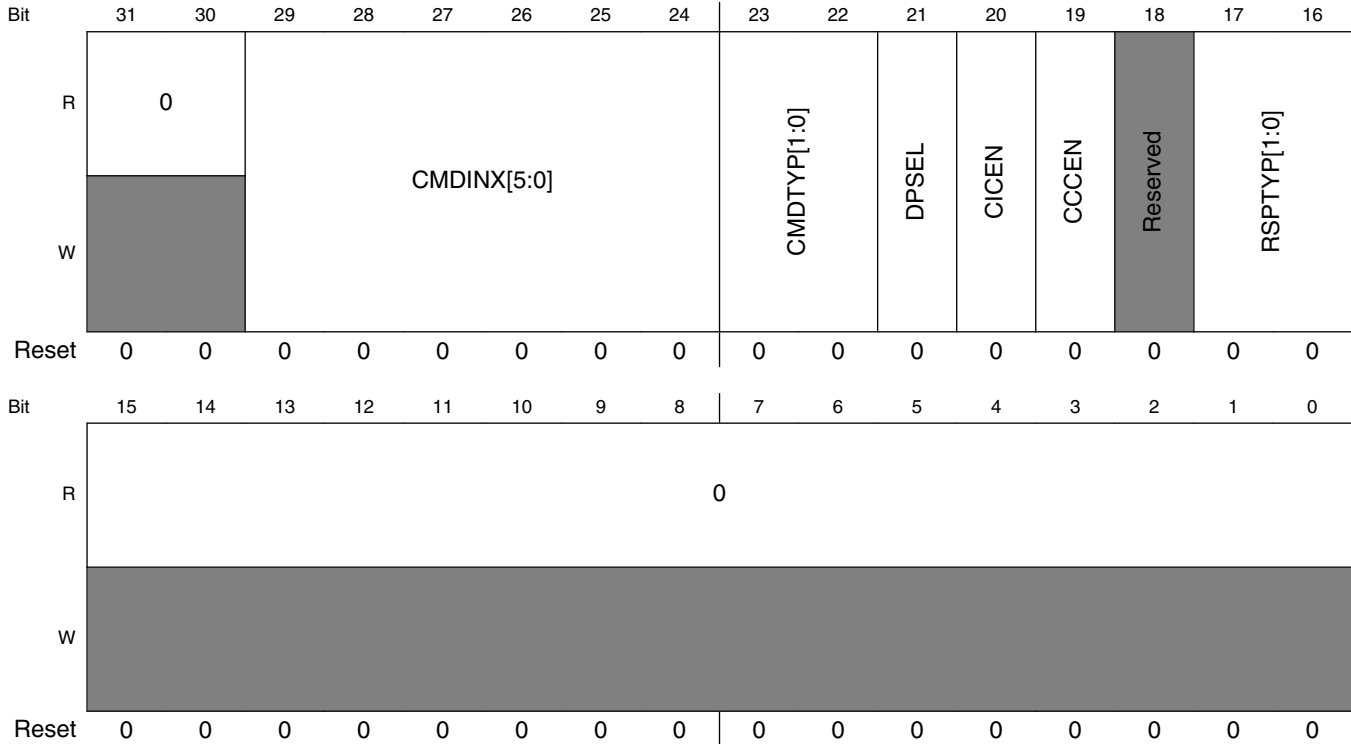
Response Type	Index Check Enable	CRC Check Enable	Name of Response Type
00	0	0	No Response
01	0	1	R2
10	0	0	R3,R4
10	1	1	R1,R5,R6
11	1	1	R1b,R5b

- In the SDIO specification, response type notation for R5b is not defined. R5 includes R5b in the SDIO specification. But R5b is defined in this specification to specify that

the uSDHC will check the busy status after receiving a response. For example, usually CMD52 is used with R5, but the I/O abort command shall be used with R5b.

- The CRC field for R3 and R4 is expected to be all 1 bits. The CRC check shall be disabled for these response types.

Address: Base address + Ch offset



**uSDHCx\_CMD\_XFR\_TYP field descriptions**

Field	Description
31–30 Reserved	This read-only field is reserved and always has the value 0.
29–24 CMDINX[5:0]	Command Index: These bits shall be set to the command number that is specified in bits 45-40 of the Command-Format in the SD Memory Card Physical Layer Specification and SDIO Card Specification.
23–22 CMDTYP[1:0]	Command Type: There are three types of special commands: Suspend, Resume and Abort. These bits shall be set to 00b for all other commands. <ul style="list-style-type: none"> <li>• Suspend Command: If the Suspend command succeeds, the uSDHC shall assume that the card bus has been released and that it is possible to issue the next command which uses the DATA line. Since the uSDHC does not monitor the content of command response, it does not know if the Suspend command succeeded or not. It is the Host Driver's responsibility to check the status of the Suspend command and send another command marked as Suspend to inform the uSDHC that a Suspend command was successfully issued. Refer to <a href="#">Suspend Resume</a> for more details. After the end bit of command is sent, the uSDHC de-asserts Read Wait for read transactions and stops checking busy for write transactions. In 4-bit mode, the interrupt cycle starts. If the Suspend command fails, the uSDHC will maintain its current state, and the Host Driver shall restart the transfer by setting the Continue Request bit in the Protocol Control register.</li> </ul>

*Table continues on the next page...*

### uSDHCx\_CMD\_XFR\_TYP field descriptions (continued)

Field	Description
	<ul style="list-style-type: none"> <li>Resume Command: The Host Driver re-starts the data transfer by restoring the registers saved before sending the Suspend Command and then sends the Resume Command. The uSDHC will check for a pending busy state before starting write transfers.</li> <li>Abort Command: If this command is set when executing a read transfer, the uSDHC will stop reads to the buffer. If this command is set when executing a write transfer, the uSDHC will stop driving the DATA line. After issuing the Abort command, the Host Driver should issue a software reset (Abort Transaction).</li> </ul> <p>11 Abort CMD12, CMD52 for writing I/O Abort in CCCR  10 Resume CMD52 for writing Function Select in CCCR  01 Suspend CMD52 for writing Bus Suspend in CCCR  00 Normal Other commands</p>
21 DPSEL	<p>Data Present Select:</p> <p>This bit is set to 1 to indicate that data is present and shall be transferred using the DATA line. It is set to 0 for the following:</p> <ul style="list-style-type: none"> <li>Commands using only the CMD line (e.g. CMD52).</li> <li>Commands with no data transfer, but using the busy signal on DATA0 line (R1b or R5b e.g. CMD38)</li> </ul> <p>Note: In resume command, this bit shall be set, and other bits in this register shall be set the same as when the transfer was initially launched. When the Write Protect switch is on, (i.e. the WPSPL bit is active as '0'), any command with a write operation will be ignored. That is to say, when this bit is set, while the DTDSEL bit is 0, writes to the register Transfer Type are ignored.</p> <p>1 Data Present  0 No Data Present</p>
20 CICEN	<p>Command Index Check Enable:</p> <p>If this bit is set to 1, the uSDHC will check the Index field in the response to see if it has the same value as the command index. If it is not, it is reported as a Command Index Error. If this bit is set to 0, the Index field is not checked.</p> <p>1 Enable  0 Disable</p>
19 CCCN	<p>Command CRC Check Enable:</p> <p>If this bit is set to 1, the uSDHC shall check the CRC field in the response. If an error is detected, it is reported as a Command CRC Error. If this bit is set to 0, the CRC field is not checked. The number of bits checked by the CRC field value changes according to the length of the response. (Refer to RSPTYP[1:0] and <a href="#">Command Transfer Type (uSDHC_CMD_XFR_TYP)</a> .)</p> <p>1 Enable  0 Disable</p>
18 -	<p>This field is reserved.  Reserved</p>
17–16 RSPTYP[1:0]	<p>Response Type Select:</p> <p>00 No Response  01 Response Length 136  10 Response Length 48  11 Response Length 48, check Busy after response</p>
15–0 Reserved	<p>This read-only field is reserved and always has the value 0.</p>



## 54.8.6 Command Response0 (uSDHCx\_CMD\_RSP0)

This register is used to store part 0 of the response bits from the card.

Address: Base address + 10h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP0[31:0]																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**uSDHCx\_CMD\_RSP0 field descriptions**

Field	Description
31–0 CMDRSP0[31:0]	Command Response 0: Refer to <a href="#">Command Response3 (uSDHC_CMD_RSP3)</a> for the mapping of command responses from the SD Bus to this register for each response type.

## 54.8.7 Command Response1 (uSDHCx\_CMD\_RSP1)

This register is used to store part 1 of the response bits from the card.

Address: Base address + 14h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP1[31:0]																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**uSDHCx\_CMD\_RSP1 field descriptions**

Field	Description
31–0 CMDRSP1[31:0]	Command Response 1: Refer to <a href="#">Command Response3 (uSDHC_CMD_RSP3)</a> for the mapping of command responses from the SD Bus to this register for each response type.

## 54.8.8 Command Response2 (uSDHCx\_CMD\_RSP2)

This register is used to store part 2 of the response bits from the card.

Address: Base address + 18h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP2[31:0]																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### uSDHCx\_CMD\_RSP2 field descriptions

Field	Description
31–0 CMDRSP2[31:0]	Command Response 2: Refer to <a href="#">Command Response3 (uSDHC_CMD_RSP3)</a> for the mapping of command responses from the SD Bus to this register for each response type.

## 54.8.9 Command Response3 (uSDHCx\_CMD\_RSP3)

This register is used to store part 3 of the response bits from the card.

The table below describes the mapping of command responses from the SD Bus to Command Response registers for each response type. In the table, R[ ] refers to a bit range within the response data as transmitted on the SD Bus.

**Table 54-59. Response Bit Definition for Each Response Type**

Response Type	Meaning of Response	Response Field	Response Register
R1,R1b (normal response)	Card Status	R[39:8]	CMDRSP0
R1b (Auto CMD12 response)	Card Status for Auto CMD12	R[39:8]	CMDRSP3
R2 (CID, CSD register)	CID/CSD register [127:8]	R[127:8]	{CMDRSP3[23:0], CMDRSP2, CMDRSP1, CMDRSP0}
R3 (OCR register)	OCR register for memory	R[39:8]	CMDRSP0
R4 (OCR register)	OCR register for I/O etc.	R[39:8]	CMDRSP0
R5, R5b	SDIO response	R[39:8]	CMDRSP0
R6 (Publish RCA)	New Published RCA[31:16] and card status[15:0]	R[39:9]	CMDRSP0

This table shows that most responses with a length of 48 (R[47:0]) have 32-bits of the response data (R[39:8]) stored in the CMDRSP0 register. Responses of type R1b (Auto CMD12 responses) have response data bits (R[39:8]) stored in the CMDRSP3 register. Responses with length 136 (R[135:0]) have 120-bits of the response data (R[127:8]) stored in the CMDRSP0, 1, 2, and 3 registers.

To be able to read the response status efficiently, the uSDHC only stores part of the response data in the Command Response registers. This enables the Host Driver to efficiently read 32-bits of response data in one read cycle on a 32-bit bus system. Parts of the response, the Index field and the CRC, are checked by the uSDHC (as specified by the Command Index Check Enable and the Command CRC Check Enable bits in the Transfer Type register) and generate an error interrupt if any error is detected. The bit range for the CRC check depends on the response length. If the response length is 48, the uSDHC will check R[47:1], and if the response length is 136 the uSDHC will check R[119:1].

Since the uSDHC may have a multiple block data transfer executing concurrently with a CMD\_wo\_DAT command, the uSDHC stores the Auto CMD12 response in the CMDRSP3 register. The CMD\_wo\_DAT response is stored in CMDRSP0. This allows the uSDHC to avoid overwriting the Auto CMD12 response with the CMD\_wo\_DAT and vice versa. When the uSDHC modifies part of the Command Response registers, as shown in the table above, it preserves the unmodified bits.

Address: Base address + 1Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	CMDRSP3[31:0]																															
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

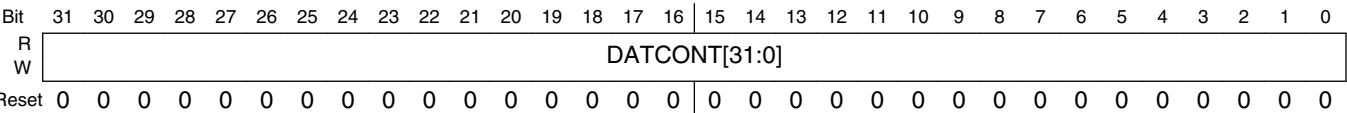
#### uSDHCx\_CMD\_RSP3 field descriptions

Field	Description
31–0 CMDRSP3[31:0]	Command Response 3: Refer to <a href="#">Command Response3 (uSDHC_CMD_RSP3)</a> for the mapping of command responses from the SD Bus to this register for each response type.

### 54.8.10 Data Buffer Access Port (uSDHCx\_DATA\_BUFF\_ACC\_PORT)

This is a 32-bit data port register used to access the internal buffer.

Address: Base address + 20h offset



uSDHCx\_DATA\_BUFF\_ACC\_PORT field descriptions

Field	Description
31–0 DATCONT[31:0]	Data Content: The Buffer Data Port register is for 32-bit data access by the ARM platform or the external DMA. When the internal DMA is enabled, any write to this register is ignored, and any read from this register will always yield 0s.

### 54.8.11 Present State (uSDHCx\_PRES\_STATE)

The Host Driver can get status of the uSDHC from this 32-bit read only register.

- The Host Driver can issue CMD0, CMD12, CMD13 (for memory) and CMD52 (for SDIO) when the DATA lines are busy during a data transfer. These commands can be issued when Command Inhibit (CMD) is set to zero. Other commands shall be issued when Command Inhibit (DATA) is set to zero. Possible changes to the SD Physical Specification may add other commands to this list in the future.
- Note: the reset value of Present State Register depend on testbench connectivity.

Address: Base address + 24h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	
R	DLSL[7:0]								CLSL	0				WPSPL	CDPL	0	CINST
W																	
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	TSCD	0		RTR	BREN	BWEN	RTA	WTA	SDOFF	PEROFF	HCKOFF	IPGOFF	SDSTB	DLA	CDIHB	CIHB
W																
Reset	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

**uSDHCx\_PRES\_STATE field descriptions**

Field	Description
31–24 DLSL[7:0]	<p>DATA[7:0] Line Signal Level:</p> <p>This status is used to check the DATA line level to recover from errors, and for debugging. This is especially useful in detecting the busy signal level from DATA0. The reset value is affected by the external pull-up/pull-down resistors. By default, the read value of this bit field after reset is 8'b11110111, when DATA3 is pulled down and the other lines are pulled up.</p> <p>DATA7: Data 7 line signal level</p> <p>DATA6: Data 6 line signal level</p> <p>DATA5: Data 5 line signal level</p> <p>DATA4: Data 4 line signal level</p> <p>DATA3: Data 3 line signal level</p> <p>DATA2: Data 2 line signal level</p> <p>DATA1: Data 1 line signal level</p> <p>DATA0: Data 0 line signal level</p>

*Table continues on the next page...*

### uSDHCx\_PRES\_STATE field descriptions (continued)

Field	Description
23 CLSL	<p>CMD Line Signal Level:</p> <p>This status is used to check the CMD line level to recover from errors, and for debugging. The reset value is affected by the external pull-up/pull-down resistor, by default, the read value of this bit after reset is 1'b1, when the command line is pulled up.</p>
22–20 Reserved	This read-only field is reserved and always has the value 0.
19 WPSP	<p>Write Protect Switch Pin Level:</p> <p>The Write Protect Switch is supported for memory and combo cards. This bit reflects the inverted value of the WP pin of the card socket. A software reset does not affect this bit. The reset value is effected by the external write protect switch. If the WP pin is not used, it should be tied low, so that the reset value of this bit is high and write is enabled.</p> <p>1 Write enabled (WP=0) 0 Write protected (WP=1)</p>
18 CDPL	<p>Card Detect Pin Level:</p> <p>This bit reflects the inverse value of the CD_B pin for the card socket. Debouncing is not performed on this bit. This bit may be valid, but is not guaranteed, because of propagation delay. Use of this bit is limited to testing since it must be debounced by software. A software reset does not effect this bit. A write to the Force Event Register does not effect this bit. The reset value is effected by the external card detection pin. This bit shows the value on the CD_B pin (i.e. when a card is inserted in the socket, it is 0 on the CD_B input, and consequently the CDPL reads 1.)</p> <p>1 Card present (CD_B=0) 0 No card present (CD_B=1)</p>
17 Reserved	This read-only field is reserved and always has the value 0.
16 CINST	<p>Card Inserted:</p> <p>This bit indicates whether a card has been inserted. The uSDHC debounces this signal so that the Host Driver will not need to wait for it to stabilize. Changing from a 0 to 1 generates a Card Insertion interrupt in the Interrupt Status register. Changing from a 1 to 0 generates a Card Removal interrupt in the Interrupt Status register. A write to the Force Event Register does not effect this bit.</p> <p>The Software Reset For All in the System Control register does not effect this bit. A software reset does not effect this bit.</p> <p>1 Card Inserted 0 Power on Reset or No Card</p>
15 TSCD	<p>Tape Select Change Done :</p> <p>This bit indicates the dealy setting is effective after write CLK_TUNE_CTRL_STATUS register.</p> <p>1 Delay cell select change is finished 0 Delay cell select change is not finished</p>
14–13 Reserved	This read-only field is reserved and always has the value 0.
12 RTR	<p>Re-Tuning Request: (only for SD3.0 SDR104 mode)</p> <p>Host Controller may request Host Driver to execute re-tuning sequence by setting this bit when the data window is shifted by temperature drift and a tuned sampling point does not have a good margin to receive correct data.</p> <p>This bit is cleared when a command is issued with setting Execute Tuning bit in MIXER_CTRL register.</p>

Table continues on the next page...

**uSDHCx\_PRES\_STATE field descriptions (continued)**

Field	Description
	<p>Changing of this bit from 0 to 1 generates Re-Tuning Event. Refer to Interrupt status registers for more detail.</p> <p>This bit isn't set to 1 if Sampling Clock Select in the MIXER_CTRL register is set to 0 (using fixed sampling clock).</p> <p>1 Sampling clock needs re-tuning 0 Fixed or well tuned sampling clock</p>
11 BREN	<p>Buffer Read Enable:</p> <p>This status bit is used for non-DMA read transfers. The uSDHC implements an internal buffer to transfer data efficiently. This read only flag indicates that valid data exists in the host side buffer. If this bit is high, valid data greater than the watermark level exist in the buffer. A change of this bit from 1 to 0 occurs when some reads from the buffer(read DATPORT(Base + 0x20)) are made and the buffer hasn't valid data greater than the watermark level. A change of this bit from 0 to 1 occurs when there is enough valid data ready in the buffer and the Buffer Read Ready interrupt has been generated and enabled.</p> <p>1 Read enable 0 Read disable</p>
10 BWEN	<p>Buffer Write Enable:</p> <p>This status bit is used for non-DMA write transfers. The uSDHC implements an internal buffer to transfer data efficiently. This read only flag indicates if space is available for write data. If this bit is 1, valid data greater than the watermark level can be written to the buffer. A change of this bit from 1 to 0 occurs when some writes to the buffer(write DATPORT(Base + 0x20)) are made and the buffer hasn't valid space greater than the watermark level. . A change of this bit from 0 to 1 occurs when the buffer can hold valid data greater than the write watermark level and the Buffer Write Ready interrupt is generated and enabled.</p> <p>1 Write enable 0 Write disable</p>
9 RTA	<p>Read Transfer Active:</p> <p>This status bit is used for detecting completion of a read transfer.</p> <p>This bit is set for either of the following conditions:</p> <ul style="list-style-type: none"> <li>• After the end bit of the read command.</li> <li>• When writing a 1 to the Continue Request bit in the Protocol Control register to restart a read transfer.</li> </ul> <p>A Transfer Complete interrupt is generated when this bit changes to 0. This bit is cleared for either of the following conditions:</p> <ul style="list-style-type: none"> <li>• When the last data block as specified by block length is transferred to the System, i.e. all data are read away from uSDHC internal buffer.</li> <li>• When all valid data blocks have been transferred from uSDHC internal buffer to the System and no current block transfers are being sent as a result of the Stop At Block Gap Request being set to 1.</li> </ul> <p>1 Transferring data 0 No valid data</p>
8 WTA	<p>Write Transfer Active:</p> <p>This status bit indicates a write transfer is active. If this bit is 0, it means no valid write data exists in the uSDHC.</p> <p>This bit is set in either of the following cases:</p>

*Table continues on the next page...*

### uSDHCx\_PRES\_STATE field descriptions (continued)

Field	Description
	<ul style="list-style-type: none"> <li>After the end bit of the write command.</li> <li>When writing 1 to the Continue Request bit in the Protocol Control register to restart a write transfer.</li> </ul> <p>This bit is cleared in either of the following cases:</p> <ul style="list-style-type: none"> <li>After getting the CRC status of the last data block as specified by the transfer count (Single and Multiple).</li> <li>After getting the CRC status of any block where data transmission is about to be stopped by a Stop At Block Gap Request.</li> </ul> <p>During a write transaction, a Block Gap Event interrupt is generated when this bit is changed to 0, as result of the Stop At Block Gap Request being set. This status is useful for the Host Driver in determining when to issue commands during Write Busy state.</p> <p>1 Transferring data 0 No valid data</p>
7 SDOFF	<p>SD Clock Gated Off Internally:</p> <p>This status bit indicates that the SD Clock is internally gated off, because of buffer over/under-run or read pause without read wait assertion, or the driver set FRC_SDCLK_ON bit is 0 to stop the SD clock in idle status. Set IPG_PERCLK_SOFT_EN and CARD_CLK_SOFT_EN to 0 also gate off SD clock. This bit is for the Host Driver to debug data transaction on the SD bus.</p> <p>1 SD Clock is gated off 0 SD Clock is active</p>
6 PEROFF	<p>ipg_perclk Gated Off Internally:</p> <p>This status bit indicates that the ipg_perclk is internally gated off. This bit is for the Host Driver to debug transaction on the SD bus. When IPG_CLK_SOFT_EN is cleared, ipg_perclk will be gated off, otherwise ipg_perclk will be always active.</p> <p>1 ipg_perclk is gated off 0 ipg_perclk is active</p>
5 HCKOFF	<p>hclk Gated Off Internally:</p> <p>This status bit indicates that the hclk is internally gated off. This bit is for the Host Driver to debug during a data transfer.</p> <p>1 hclk is gated off 0 hclk is active</p>
4 IPGOFF	<p>ipg_clk Gated Off Internally:</p> <p>This status bit indicates that the ipg_clk is internally gated off. This bit is for the Host Driver to debug.</p> <p>1 ipg_clk is gated off 0 ipg_clk is active</p>
3 SDSTB	<p>SD Clock Stable</p> <p>This status bit indicates that the internal card clock is stable. This bit is for the Host Driver to poll clock status when changing the clock frequency. It is recommended to clear FRC_SDCLK_ON bit in System Control register to remove glitches on the card clock when the frequency is changing.</p> <p><i>Before changing clock divisor value(SDCLKFS or DVS), Host Driver should make sure the SDSTB bit is high.</i></p>

Table continues on the next page...



**uSDHCx\_PRES\_STATE field descriptions (continued)**

Field	Description
	<p>1 clock is stable</p> <p>0 clock is changing frequency and not stable</p>
2 DLA	<p><b>Data Line Active</b></p> <p>This status bit indicates whether one of the DATA lines on the SD Bus is in use.</p> <p>In the case of read transactions:</p> <p>This status indicates if a read transfer is executing on the SD Bus. Changes in this value from 1 to 0, between data blocks, generates a Block Gap Event interrupt in the Interrupt Status register.</p> <p>This bit will be set in either of the following cases:</p> <ul style="list-style-type: none"> <li>• After the end bit of the read command.</li> <li>• When writing a 1 to the Continue Request bit in the Protocol Control register to restart a read transfer.</li> </ul> <p>This bit will be cleared in either of the following cases:</p> <p>(1) When the end bit of the last data block is sent from the SD Bus to the uSDHC.</p> <p>(2) When the Read Wait state is stopped by a Suspend command and the DATA2 line is released.</p> <p>The uSDHC will wait at the next block gap by driving Read Wait at the start of the interrupt cycle. If the Read Wait signal is already driven (data buffer cannot receive data), the uSDHC can wait for a current block gap by continuing to drive the Read Wait signal. It is necessary to support Read Wait in order to use the suspend / resume function. This bit will remain 1 during Read Wait.</p> <p>In the case of write transactions:</p> <p>This status indicates that a write transfer is executing on the SD Bus. Changes in this value from 1 to 0 generate a Transfer Complete interrupt in the Interrupt Status register.</p> <p>This bit will be set in either of the following cases:</p> <ul style="list-style-type: none"> <li>• After the end bit of the write command.</li> <li>• When writing to 1 to the Continue Request bit in the Protocol Control register to continue a write transfer.</li> </ul> <p>This bit will be cleared in either of the following cases:</p> <ul style="list-style-type: none"> <li>• When the SD card releases Write Busy of the last data block, the uSDHC will also detect if the output is not busy. If the SD card does not drive the busy signal after the CRC status is received, the uSDHC shall assume the card drive "Not Busy".</li> <li>• When the SD card releases write busy, prior to waiting for write transfer, and as a result of a Stop At Block Gap Request.</li> </ul> <p>In the case of command with busy pending:</p> <p>This status indicates that a busy state follows the command and the data line is in use. This bit will be cleared when the DATA0 line is released.</p> <p>1 DATA Line Active</p> <p>0 DATA Line Inactive</p>
1 CDIHB	<p><b>Command Inhibit (DATA):</b></p> <p>This status bit is generated if either the DAT Line Active or the Read Transfer Active is set to 1. If this bit is 0, it indicates that the uSDHC can issue the next SD/MMC Command. Commands with a busy signal belong to Command Inhibit (DATA) (e.g. R1b, R5b type). Except in the case when the command busy is finished, changing from 1 to 0 generates a Transfer Complete interrupt in the Interrupt Status register.</p> <p>Note: The SD Host Driver can save registers for a suspend transaction after this bit has changed from 1 to 0.</p>

*Table continues on the next page...*

### uSDHCx\_PRES\_STATE field descriptions (continued)

Field	Description
	1 Cannot issue command which uses the DATA line 0 Can issue command which uses the DATA line
0 CIHB	<p>Command Inhibit (CMD):</p> <p>If this status bit is 0, it indicates that the CMD line is not in use and the uSDHC can issue a SD/MMC Command using the CMD line.</p> <p>This bit is set also immediately after the Transfer Type register is written. This bit is cleared when the command response is received. Even if the Command Inhibit (DATA) is set to 1, Commands using only the CMD line can be issued if this bit is 0. Changing from 1 to 0 generates a Command Complete interrupt in the Interrupt Status register. If the uSDHC cannot issue the command because of a command conflict error (Refer to Command CRC Error) or because of a Command Not Issued By Auto CMD12 Error, this bit will remain 1 and the Command Complete is not set. The Status of issuing an Auto CMD12 does not show on this bit.</p> <p>1 Cannot issue command 0 Can issue command using only CMD line</p>

## 54.8.12 Protocol Control (uSDHCx\_PROT\_CTRL)

There are three cases to restart the transfer after stop at the block gap. Which case is appropriate depends on whether the uSDHC issues a Suspend command or the SD card accepts the Suspend command.

1. If the Host Driver does not issue a Suspend command, the Continue Request shall be used to restart the transfer.
2. If the Host Driver issues a Suspend command and the SD card accepts it, a Resume command shall be used to restart the transfer.
3. If the Host Driver issues a Suspend command and the SD card does not accept it, the Continue Request shall be used to restart the transfer.

Any time Stop At Block Gap Request stops the data transfer, the Host Driver shall wait for a Transfer Complete (in the Interrupt Status register), before attempting to restart the transfer. When restarting the data transfer by Continue Request, the Host Driver shall clear the Stop At Block Gap Request before or simultaneously.

Address: Base address + 28h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	-	NON_EXACT_BLK_RD	BURST_LEN_EN			WECRM	WECINS	WECINT	-			RD_DONE_NO_8CLK	IABG	RWCTL	CREQ	SABGREQ
W	-	NON_EXACT_BLK_RD	BURST_LEN_EN			WECRM	WECINS	WECINT	-			RD_DONE_NO_8CLK	IABG	RWCTL	CREQ	SABGREQ
Reset	0	0	0	0	1	0	0	0	1	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0						DMASEL		CDSS	CDTL	EMODE		D3CD	DTW[1:0]		LCTL
W																
Reset	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0

**uSDHCx\_PROT\_CTRL field descriptions**

Field	Description
31 -	Reserved. Always write as 0
30 NON_EXACT_BLK_RD	Current block read is non-exact block read. It's only used for SDIO.  1 The block read is non-exact block read. Host driver needs to issue abort command to terminate this multi-block read. 0 The block read is exact block read. Host driver doesn't need to issue abort command to terminate this multi-block read.
29-27 BURST_LEN_EN	BURST length enable for INCR, INCR4/INCR8/INCR16, INCR4-WRAP/INCR8-WRAP/INCR16-WRAP  This is used to enable/disable the burst length for the external AHB2AXI bridge. It's useful especially for INCR transfer because without burst length indicator, the AHB2AXI bridge doesn't know the burst length in advance. Without burst length indicator, AHB INCR transfers can only be converted to SINGLES on the AXI side.  xx1 Burst length is enabled for INCR x1x Burst length is enabled for INCR4/INCR8/INCR16 1xx Burst length is enabled for INCR4-WRAP/INCR8-WRAP/INCR16-WRAP
26 WECRM	Wakeup Event Enable On SD Card Removal:  This bit enables a wakeup event, via a Card Removal, in the Interrupt Status register. FN_WUS (Wake Up Support) in CIS does not effect this bit. When this bit is set, the Card Removal Status and the uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active in order to assert the Card Removal Status and the uSDHC interrupt.  1 Enable 0 Disable
25 WECINS	Wakeup Event Enable On SD Card Insertion:  This bit enables a wakeup event, via a Card Insertion, in the Interrupt Status register. FN_WUS (Wake Up Support) in CIS does not effect this bit. When this bit is set, the Card Insertion Status and the uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active in order to assert the Card Insertion Status and the uSDHC interrupt.  1 Enable 0 Disable
24 WECINT	Wakeup Event Enable On Card Interrupt:  This bit enables a wakeup event, via a Card Interrupt, in the Interrupt Status register. This bit can be set to 1 if FN_WUS (Wake Up Support) in CIS is set to 1. When this bit is set, the Card Interrupt Status and the uSDHC interrupt can be asserted without CLK toggling. When the wakeup feature is not enabled, the CLK must be active in order to assert the Card Interrupt Status and the uSDHC interrupt.

*Table continues on the next page...*

### uSDHCx\_PROT\_CTRL field descriptions (continued)

Field	Description
	1 Enable 0 Disable
23–21 -	Reserved. Always write as 3'b100
20 RD_DONE_NO_8CLK	<p><i>Read done no 8 clock:</i></p> <p><i>According to the SD/MMC spec, for read data transaction, 8 clocks are needed after the end bit of the last data block. So, by default(RD_DONE_NO_8CLK=0), 8 clocks will be active after the end bit of the last read data transaction.</i></p> <p><i>However, this 8 clocks should not be active if user wants to use stop at block gap(include the auto stop at block gap in boot mode) feature for read and the RWCTL bit(bit18) is not enabled. In this case, software should set RD_DONE_NO_8CLK to avoid this 8 clocks. Otherwise, the device may send extra data to uSDHC while uSDHC ignores these data.</i></p> <p><i>In a summary, this bit should be set only if the use case needs to use stop at block gap feature while the device can't support the read wait feature.</i></p>
19 IABG	<p>Interrupt At Block Gap:</p> <p>This bit is valid only in 4-bit mode, of the SDIO card, and selects a sample point in the interrupt cycle. Setting to 1 enables interrupt detection at the block gap for a multiple block transfer. Setting to 0 disables interrupt detection during a multiple block transfer. If the SDIO card can't signal an interrupt during a multiple block transfer, this bit should be set to 0 to avoid an inadvertent interrupt. When the Host Driver detects an SDIO card insertion, it shall set this bit according to the CCCR of the card.</p> <p>1 Enabled 0 Disabled</p>
18 RWCTL	<p>Read Wait Control:</p> <p>The read wait function is optional for SDIO cards. If the card supports read wait, set this bit to enable use of the read wait protocol to stop read data using the DATA2 line. Otherwise the uSDHC has to stop the SD Clock to hold read data, which restricts commands generation. When the Host Driver detects an SDIO card insertion, it shall set this bit according to the CCCR of the card. If the card does not support read wait, this bit shall never be set to 1, otherwise DATA line conflicts may occur. If this bit is set to 0, stop at block gap during read operation is also supported, but the uSDHC will stop the SD Clock to pause reading operation.</p> <p>1 Enable Read Wait Control, and assert Read Wait without stopping SD Clock at block gap when SABGREQ bit is set 0 Disable Read Wait Control, and stop SD Clock at block gap when SABGREQ bit is set</p>
17 CREQ	<p>Continue Request:</p> <p>This bit is used to restart a transaction which was stopped using the Stop At Block Gap Request. When a Suspend operation is not accepted by the card, it is also by setting this bit to restart the paused transfer. To cancel stop at the block gap, set Stop At Block Gap Request to 0 and set this bit to 1 to restart the transfer.</p> <p>The uSDHC automatically clears this bit, therefore it is not necessary for the Host Driver to set this bit to 0. If both Stop At Block Gap Request and this bit are 1, the continue request is ignored.</p> <p>1 Restart 0 No effect</p>
16 SABGREQ	Stop At Block Gap Request:

Table continues on the next page...

**uSDHCx\_PROT\_CTRL field descriptions (continued)**

Field	Description
	<p>This bit is used to stop executing a transaction at the next block gap for both DMA and non-DMA transfers. Until the Transfer Complete is set to 1, indicating a transfer completion, the Host Driver shall leave this bit set to 1. Clearing both the Stop At Block Gap Request and Continue Request does not cause the transaction to restart. Read Wait is used to stop the read transaction at the block gap. The uSDHC will honor the Stop At Block Gap Request for write transfers, but for read transfers it requires that the SDIO card support Read Wait. Therefore, the Host Driver shall not set this bit during read transfers unless the SDIO card supports Read Wait and has set the Read Wait Control to 1, otherwise the uSDHC will stop the SD bus clock to pause the read operation during block gap. In the case of write transfers in which the Host Driver writes data to the Data Port register, the Host Driver shall set this bit after all block data is written. If this bit is set to 1, the Host Driver shall not write data to the Data Port register after a block is sent. Once this bit is set, the Host Driver shall not clear this bit before the Transfer Complete bit in Interrupt Status Register is set, otherwise the uSDHCs behavior is undefined.</p> <p>This bit effects Read Transfer Active, Write Transfer Active, DATA Line Active and Command Inhibit (DATA) in the Present State register.</p> <p>1 Stop 0 Transfer</p>
15–10 Reserved	This read-only field is reserved and always has the value 0.
9–8 DMASEL	<p>DMA Select:</p> <p>This field is valid while DMA (SDMA or ADMA) is enabled and selects the DMA operation.</p> <p>00 No DMA or Simple DMA is selected 01 ADMA1 is selected 10 ADMA2 is selected 11 reserved</p>
7 CDSS	<p>Card Detect Signal Selection:</p> <p>This bit selects the source for the card detection.</p> <p>1 Card Detection Test Level is selected (for test purpose) 0 Card Detection Level is selected (for normal purpose)</p>
6 CDTL	<p>Card Detect Test Level:</p> <p>This bit is enabled while the Card Detection Signal Selection is set to 1 and it indicates card insertion.</p> <p>1 Card Detect Test Level is 1, card inserted 0 Card Detect Test Level is 0, no card inserted</p>
5–4 EMODE	<p>Endian Mode:</p> <p>The uSDHC supports all three endian modes in data transfer. Refer to <a href="#">Data Buffer</a> for more details.</p> <p>00 Big Endian Mode 01 Half Word Big Endian Mode 10 Little Endian Mode 11 Reserved</p>
3 D3CD	<p>DATA3 as Card Detection Pin:</p> <p>If this bit is set, DATA3 should be pulled down to act as a card detection pin. Be cautious when using this feature, because DATA3 is also a chip-select for the SPI mode. A pull-down on this pin and CMD0 may set the card into the SPI mode, which the uSDHC does not support.</p>

*Table continues on the next page...*

**uSDHCx\_PROT\_CTRL field descriptions (continued)**

Field	Description
	1 DATA3 as Card Detection Pin 0 DATA3 does not monitor Card Insertion
2–1 DTW[1:0]	Data Transfer Width: This bit selects the data width of the SD bus for a data transfer. The Host Driver shall set it to match the data width of the card. Possible Data transfer Width is 1-bit, 4-bits or 8-bits.  10 8-bit mode 01 4-bit mode 00 1-bit mode 11 Reserved
0 LCTL	LED Control: This bit, fully controlled by the Host Driver, is used to caution the user not to remove the card while the card is being accessed. If the software is going to issue multiple SD commands, this bit can be set during all these transactions. It is not necessary to change for each transaction. When the software issues multiple SD commands, setting the bit once before the first command is sufficient: it is not necessary to reset the bit between commands.  1 LED on 0 LED off

### 54.8.13 System Control (uSDHCx\_SYS\_CTRL)

Address: Base address + 2Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			RSTT	INITA	0	0	0	IPP_ RST_ N	-	0	DTCV				
W						RSTD	RSTC	RSTA								
Reset	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	SDCLKFS								DVS[3:0]				-			
W																
Reset	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

**uSDHCx\_SYS\_CTRL field descriptions**

Field	Description
31–29 Reserved	This read-only field is reserved and always has the value 0.
28 RSTT	Reset Tuning: When set this bit to 1, it will reset tuning circuit. After tuning circuits are reset, bit value is 0. Clearing execute_tuning bit in AUTOCMD12_ERR_STATUS will also set this bit to 1 to reset tuning circuit
27 INITA	Initialization Active: When this bit is set, 80 SD-Clocks are sent to the card. After the 80 clocks are sent, this bit is self cleared. This bit is very useful during the card power-up period when 74 SD-Clocks are needed and the clock auto gating feature is enabled. Writing 1 to this bit when this bit is already 1 has no effect. Writing 0 to this bit at any time has no effect. When either of the CIHB and CDIHB bits in the Present State Register are set, writing 1 to this bit is ignored (i.e. when command line or data lines are active, write to this bit is not allowed). On the otherhand, when this bit is set, i.e., during intialization active period, it is allowed to issue command, and the command bit stream will appear on the CMD pad after all 80 clock cycles are done. So when this command ends, the driver can make sure the 80 clock cycles are sent out. This is very useful when the driver needs send 80 cycles to the card and does not want to wait till this bit is self cleared.
26 RSTD	Software Reset For DATA Line:

*Table continues on the next page...*

### uSDHCx\_SYS\_CTRL field descriptions (continued)

Field	Description
	<p>Only part of the data circuit is reset. DMA circuit is also reset. After this bit is set, SW should wait for slef-clear.</p> <p>The following registers and bits are cleared by this bit:</p> <ul style="list-style-type: none"> <li>• Data Port register</li> <li>• Buffer is cleared and initialized.</li> <li>• Present State register</li> <li>• Buffer Read Enable</li> <li>• Buffer Write Enable</li> <li>• Read Transfer Active</li> <li>• Write Transfer Active</li> <li>• DATA Line Active</li> <li>• Command Inhibit (DATA) Protocol Control register</li> <li>• Continue Request</li> <li>• Stop At Block Gap Request Interrupt Status register</li> <li>• Buffer Read Ready</li> <li>• Buffer Write Ready</li> <li>• DMA Interrupt</li> <li>• Block Gap Event</li> <li>• Transfer Complete</li> </ul> <p>1    Reset 0    No Reset</p>
25 RSTC	<p>Software Reset For CMD Line:</p> <p>Only part of the command circuit is reset. After this bit is set, SW should wait for slef-clear.</p> <p>The following registers and bits are cleared by this bit:</p> <ul style="list-style-type: none"> <li>• Present State register Command Inhibit (CMD)</li> <li>• Interrupt Status register Command Complete</li> </ul> <p>1    Reset 0    No Reset</p>
24 RSTA	<p>Software Reset For ALL:</p> <p>This reset effects the entire Host Controller except for the card detection circuit. Register bits of type ROC, RW, RW1C, RWAC are cleared. During its initialization, the Host Driver shall set this bit to 1 to reset the uSDHC. The uSDHC shall reset this bit to 0 when the capabilities registers are valid and the Host Driver can read them. Additional use of Software Reset For All does not affect the value of the Capabilities registers. After this bit is set, it is recommended that the Host Driver reset the external card and re-initialize it. After this bit is set, SW should wait for slef-clear.</p> <p>In tuning process, after every CMD19 is finished, this bit will be set to resett the uSDHC.</p> <p>1    Reset 0    No Reset</p>
23 IPP_RST_N	This register's value will be output to CARD from pad directly for hardware reset of the card if the card supports this feature.
22 -	Reserved
21–20 Reserved	This read-only field is reserved and always has the value 0.
19–16 DTCOV	Data Timeout Counter Value:

Table continues on the next page...



**uSDHCx\_SYS\_CTRL field descriptions (continued)**

Field	Description
	<p>This value determines the interval by which DAT line timeouts are detected. Refer to the Data Timeout Error bit in the Interrupt Status register for information on factors that dictate time-out generation. Time-out clock frequency will be generated by dividing the base clock SDCLK value by this value.</p> <p>The Host Driver can clear the Data Timeout Error Status Enable (in the Interrupt Status Enable register) to prevent inadvertent time-out events.</p> <p>1111 SDCLK x 2<sup>28</sup>  1110 SDCLK x 2<sup>27</sup>  0001 SDCLK x 2<sup>14</sup>  0000 SDCLK x 2<sup>13</sup></p>
15–8 SDCLKFS	<p>SDCLK Frequency Select:</p> <p>This register is used to select the frequency of the SDCLK pin. The frequency is not programmed directly, rather this register holds the prescaler (this register) and divisor (next register) of the Base Clock Frequency register.</p> <p><i>In Single Data Rate mode(DDR_EN bit of MIXERCTRL is '0')</i></p> <p>Only the following settings are allowed:</p> <p>80h) Base clock divided by 256  40h) Base clock divided by 128  20h) Base clock divided by 64  10h) Base clock divided by 32  08h) Base clock divided by 16  04h) Base clock divided by 8  02h) Base clock divided by 4  01h) Base clock divided by 2  00h) Base clock divided by 1</p> <p><i>While in Dual Data Rate mode(DDR_EN bit of MIXERCTRL is '1')</i></p> <p>Only the following settings are allowed:</p> <p>80h) Base clock divided by 512  40h) Base clock divided by 256  20h) Base clock divided by 128  10h) Base clock divided by 64  08h) Base clock divided by 32  04h) Base clock divided by 16  02h) Base clock divided by 8  01h) Base clock divided by 4  00h) Base clock divided by 2</p> <p><i>When S/W changes the DDR_EN bit, SDCLKFS may need to be changed also!</i></p> <p>In Single Data Rate mode, setting 00h bypasses the frequency prescaler of the SD Clock.</p> <p>Multiple bits must not be set, or the behavior of this prescaler is undefined. The two default divider values can be calculated by the frequency of ipg_perclk and the following Divisor bits.</p> <p>The frequency of SDCLK is set by the following formula:</p>

*Table continues on the next page...*

**uSDHCx\_SYS\_CTRL field descriptions (continued)**

Field	Description										
	<p>Clock Frequency = (Base Clock) / (prescaler x divisor)</p> <p>For example, in Single Data Rate mode, if the Base Clock Frequency is 96 MHz, and the target frequency is 25 MHz, then choosing the prescaler value of 01h and divisor value of 1h will yield 24 MHz, which is the nearest frequency less than or equal to the target. Similarly, to approach a clock value of 400 kHz, the prescaler value of 08h and divisor value of eh yields the exact clock value of 400 kHz.</p> <p>The reset value of this bit field is 80h, so if the input Base Clock (ipg_perclk) is about 96 MHz, the default SD Clock after reset is 375 kHz.</p> <p>According to the SD Physical Specification Version 1.1 and the SDIO Card Specification Version 1.2, the maximum SD Clock frequency is 50 MHz and shall never exceed this limit.</p> <p><i>Before changing clock divisor value(SDCLKFS or DVS), Host Driver should make sure the SDSTB bit is high.</i></p> <p><i>If setting SDCLKFS and DVS can generate same clock frequency,(For example, in SDR mode, SDCLKFS = 01h is same as DVS = 01h.) SDCLKFS is highly recommended.</i></p>										
7-4 DVS[3:0]	<p>Divisor:</p> <p>This register is used to provide a more exact divisor to generate the desired SD clock frequency. Note the divider can even support odd divisors without deterioration of duty cycle.</p> <p><i>Before changing clock divisor value(SDCLKFS or DVS), Host Driver should make sure the SDSTB bit is high.</i></p> <p>The setting are as following:</p> <table> <tr> <td>0000</td><td>Divide-by-1</td></tr> <tr> <td>0001</td><td>Divide-by-2</td></tr> <tr> <td>.....</td><td></td></tr> <tr> <td>1110</td><td>Divide-by-15</td></tr> <tr> <td>1111</td><td>Divide-by-16</td></tr> </table>	0000	Divide-by-1	0001	Divide-by-2	.....		1110	Divide-by-15	1111	Divide-by-16
0000	Divide-by-1										
0001	Divide-by-2										
.....											
1110	Divide-by-15										
1111	Divide-by-16										
3-0 -	Reserved. Always write as 1.										

**54.8.14 Interrupt Status (uSDHCx\_INT\_STATUS)**

An interrupt is generated when the Normal Interrupt Signal Enable is enabled and at least one of the status bits is set to 1. For all bits, writing 1 to a bit clears it; writing to 0 keeps the bit unchanged. More than one status can be cleared with a single register write. For Card Interrupt, before writing 1 to clear, it is required that the card stops asserting the interrupt, meaning that when the Card Driver services the interrupt condition, otherwise the CINT bit will be asserted again.

The table below shows the relationship between the Command Timeout Error and the Command Complete.

**Table 54-65. uSDHC Status for Command Timeout Error/Command Complete Bit Combinations**

Command Complete	Command Timeout Error	Meaning of the Status
0	0	X
X	1	Response not received within 64 SDCLK cycles
1	0	Response received

The table below shows the relationship between the Transfer Complete and the Data Timeout Error.

**Table 54-66. uSDHC Status for Data Timeout Error/Transfer Complete Bit Combinations**

Transfer Complete	Data Timeout Error	Meaning of the Status
0	0	X
0	1	Timeout occurred during transfer
1	X	Data Transfer Complete

The table below shows the relationship between the Command CRC Error and Command Timeout Error.

**Table 54-67. uSDHC Status for Command CRC Error/Command Timeout Error Bit Combinations**

Command Complete	Command Timeout Error	Meaning of the Status
0	0	No error
0	1	Response Timeout Error
1	0	Response CRC Error
1	1	CMD line conflict

Address: Base address + 30h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			DMAE	0	TNE	0	AC12E	0	DEBE	DCE	DTOE	CIE	CEBE	CCE	CTOE
W				w1c		w1c		w1c		w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

## uSDHC Memory Map/Register Definition

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	TP	0	RTE	0			CINT	CRM	CINS	BRR	BWR	DINT	BGE	TC	CC
W		w1c		w1c				w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c	w1c
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### uSDHCx\_INT\_STATUS field descriptions

Field	Description
31–29 Reserved	This read-only field is reserved and always has the value 0.
28 DMAE	<p>DMA Error:</p> <p>Occurs when an Internal DMA transfer has failed. This bit is set to 1, when some error occurs in the data transfer. This error can be caused by either Simple DMA or ADMA, depending on which DMA is in use. The value in DMA System Address register is the next fetch address where the error occurs. Since any error corrupts the whole data block, the Host Driver shall re-start the transfer from the corrupted block boundary. The address of the block boundary can be calculated either from the current DS_ADDR value or from the remaining number of blocks and the block size.</p> <p>1 Error 0 No Error</p>
27 Reserved	This read-only field is reserved and always has the value 0.
26 TNE	<p>Tuning Error: (only for SD3.0 SDR104 mode)</p> <p>This bit is set when an unrecoverable error is detected in a tuning circuit. By detecting Tuning Error, Host Driver needs to abort a command executing and perform tuning.</p>
25 Reserved	This read-only field is reserved and always has the value 0.
24 AC12E	<p>Auto CMD12 Error:</p> <p>Occurs when detecting that one of the bits in the Auto CMD12 Error Status register has changed from 0 to 1. This bit is set to 1, not only when the errors in Auto CMD12 occur, but also when the Auto CMD12 is not executed due to the previous command error.</p> <p>1 Error 0 No Error</p>
23 Reserved	This read-only field is reserved and always has the value 0.
22 DEBE	<p>Data End Bit Error:</p> <p>Occurs either when detecting 0 at the end bit position of read data, which uses the DATA line, or at the end bit position of the CRC.</p> <p>This bit will be not asserted in tuning process.</p> <p>1 Error 0 No Error</p>
21 DCE	Data CRC Error:

Table continues on the next page...

**uSDHCx\_INT\_STATUS field descriptions (continued)**

Field	Description
	<p>Occurs when detecting a CRC error when transferring read data, which uses the DATA line, or when detecting the Write CRC status having a value other than 010.</p> <p>This bit will be not asserted in tuning process.</p> <p>1 Error 0 No Error</p>
20 DTOE	<p>Data Timeout Error:</p> <p>Occurs when detecting one of following time-out conditions.</p> <ul style="list-style-type: none"> <li>• Busy time-out for R1b,R5b type</li> <li>• Busy time-out after Write CRC status</li> <li>• Read Data time-out.</li> </ul> <p>This bit will be not asserted in tuning process.</p> <p>1 Time out 0 No Error</p>
19 CIE	<p>Command Index Error:</p> <p>Occurs if a Command Index error occurs in the command response.</p> <p>This bit will be not asserted in tuning process.</p> <p>1 Error 0 No Error</p>
18 CEBE	<p>Command End Bit Error:</p> <p>Occurs when detecting that the end bit of a command response is 0.</p> <p>This bit will be not asserted in tuning process.</p> <p>1 End Bit Error Generated 0 No Error</p>
17 CCE	<p>Command CRC Error:</p> <p>Command CRC Error is generated in two cases.</p> <ul style="list-style-type: none"> <li>• If a response is returned and the Command Timeout Error is set to 0 (indicating no time-out), this bit is set when detecting a CRC error in the command response.</li> <li>• The uSDHC detects a CMD line conflict by monitoring the CMD line when a command is issued. If the uSDHC drives the CMD line to 1, but detects 0 on the CMD line at the next SDCLK edge, then the uSDHC shall abort the command (Stop driving CMD line) and set this bit to 1. The Command Timeout Error shall also be set to 1 to distinguish CMD line conflict.</li> </ul> <p>This bit will be not asserted in tuning process.</p> <p>1 CRC Error Generated. 0 No Error</p>
16 CTOE	<p>Command Timeout Error:</p> <p>Occurs only if no response is returned within 64 SDCLK cycles from the end bit of the command. If the uSDHC detects a CMD line conflict, in which case a Command CRC Error shall also be set (as shown in <a href="#">Interrupt Status (uSDHCx_INT_STATUS)</a> ), this bit shall be set without waiting for 64 SDCLK cycles. This is because the command will be aborted by the uSDHC.</p> <p>This bit will be not asserted in tuning process.</p>

*Table continues on the next page...*

### uSDHCx\_INT\_STATUS field descriptions (continued)

Field	Description
	1 Time out 0 No Error
15 Reserved	This read-only field is reserved and always has the value 0.
14 TP	Tuning Pass:(only for SD3.0 SDR104 mode) Current CMD19 transfer is done successfully. That is, current sampling point is correct.
13 Reserved	This read-only field is reserved and always has the value 0.
12 RTE	Re-Tuning Event: (only for SD3.0 SDR104 mode) This status is set if Re-Tuning Request in the Present State register changes from 0 to 1. Host Controller requests Host Driver to perform re-tuning for next data transfer. Current data transfer (not large block count) can be completed without re-tuning.  1 Re-Tuning should be performed 0 Re-Tuning is not required
11–9 Reserved	This read-only field is reserved and always has the value 0.
8 CINT	Card Interrupt: This status bit is set when an interrupt signal is detected from the external card. In 1-bit mode, the uSDHC will detect the Card Interrupt without the SD Clock to support wakeup. In 4-bit mode, the card interrupt signal is sampled during the interrupt cycle, so the interrupt from card can only be sampled during interrupt cycle, introducing some delay between the interrupt signal from the SDIO card and the interrupt to the Host System. Writing this bit to 1 can clear this bit, but as the interrupt source from the SDIO card does not clear, this bit is set again. In order to clear this bit, it is required to reset the interrupt source from the external card followed by a writing 1 to this bit.  When this status has been set, and the Host Driver needs to service this interrupt, the Card Interrupt Signal Enable in the Interrupt Signal Enable register should be 0 to stop driving the interrupt signal to the Host System. After completion of the card interrupt service (It should reset the interrupt sources in the SDIO card and the interrupt signal may not be asserted), write 1 to clear this bit, set the Card Interrupt Signal Enable to 1, and start sampling the interrupt signal again.  1 Generate Card Interrupt 0 No Card Interrupt
7 CRM	Card Removal: This status bit is set if the Card Inserted bit in the Present State register changes from 1 to 0. When the Host Driver writes this bit to 1 to clear this status, the status of the Card Inserted in the Present State register should be confirmed. Because the card state may possibly be changed when the Host Driver clears this bit and the interrupt event may not be generated. When this bit is cleared, it will be set again if no card is inserted. In order to leave it cleared, clear the Card Removal Status Enable bit in Interrupt Status Enable register.  1 Card removed 0 Card state unstable or inserted
6 CINS	Card Insertion: This status bit is set if the Card Inserted bit in the Present State register changes from 0 to 1. When the Host Driver writes this bit to 1 to clear this status, the status of the Card Inserted in the Present State register should be confirmed. Because the card state may possibly be changed when the Host Driver

Table continues on the next page...

**uSDHCx\_INT\_STATUS field descriptions (continued)**

Field	Description
	<p>clears this bit and the interrupt event may not be generated. When this bit is cleared, it will be set again if a card is inserted. In order to leave it cleared, clear the Card Inserted Status Enable bit in Interrupt Status Enable register.</p> <p>1 Card inserted 0 Card state unstable or removed</p>
5 BRR	<p>Buffer Read Ready:</p> <p>This status bit is set if the Buffer Read Enable bit, in the Present State register, changes from 0 to 1. Refer to the Buffer Read Enable bit in the Present State register for additional information.</p> <p>This bit indicates that cmd19 is finished in tuning process.</p> <p>1 Ready to read buffer 0 Not ready to read buffer</p>
4 BWR	<p>Buffer Write Ready:</p> <p>This status bit is set if the Buffer Write Enable bit, in the Present State register, changes from 0 to 1. Refer to the Buffer Write Enable bit in the Present State register for additional information.</p> <p>1 Ready to write buffer: 0 Not ready to write buffer</p>
3 DINT	<p>DMA Interrupt:</p> <p>Occurs only when the internal DMA finishes the data transfer successfully. Whenever errors occur during data transfer, this bit will not be set. Instead, the DMAE bit will be set. Either Simple DMA or ADMA finishes data transferring, this bit will be set.</p> <p>1 DMA Interrupt is generated 0 No DMA Interrupt</p>
2 BGE	<p>Block Gap Event:</p> <p>If the Stop At Block Gap Request bit in the Protocol Control register is set, this bit is set when a read or write transaction is stopped at a block gap. If Stop At Block Gap Request is not set to 1, this bit is not set to 1.</p> <p>In the case of a Read Transaction: This bit is set at the falling edge of the DATA Line Active Status (When the transaction is stopped at SD Bus timing). The Read Wait must be supported in order to use this function.</p> <p>In the case of Write Transaction: This bit is set at the falling edge of Write Transfer Active Status (After getting CRC status at SD Bus timing).</p> <p>1 Transaction stopped at block gap 0 No block gap event</p>
1 TC	<p>Transfer Complete:</p> <p>This bit is set when a read or write transfer is completed.</p> <p>In the case of a Read Transaction: This bit is set at the falling edge of the Read Transfer Active Status. There are two cases in which this interrupt is generated. The first is when a data transfer is completed as specified by the data length (after the last data has been read to the Host System). The second is when data has stopped at the block gap and completed the data transfer by setting the Stop At Block Gap Request bit in the Protocol Control register (after valid data has been read to the Host System).</p> <p>In the case of a Write Transaction: This bit is set at the falling edge of the DATA Line Active Status. There are two cases in which this interrupt is generated. The first is when the last data is written to the SD card as specified by the data length and the busy signal is released. The second is when data transfers are</p>

*Table continues on the next page...*

### uSDHCx\_INT\_STATUS field descriptions (continued)

Field	Description
	<p>stopped at the block gap, by setting the Stop At Block Gap Request bit in the Protocol Control register, and the data transfers are completed. (after valid data is written to the SD card and the busy signal released).</p> <p>This bit will be not asserted in tuning process.</p> <p>1 Transfer complete 0 Transfer not complete</p>
0 CC	<p>Command Complete:</p> <p>This bit is set when you receive the end bit of the command response (except Auto CMD12). Refer to the Command Inhibit (CMD) in the Present State register.</p> <p>This bit will be not asserted in tuning process.</p> <p>1 Command complete 0 Command not complete</p>

### 54.8.15 Interrupt Status Enable (uSDHCx\_INT\_STATUS\_EN)

Setting the bits in this register to 1 enables the corresponding Interrupt Status to be set by the specified event. If any bit is cleared, the corresponding Interrupt Status bit is also cleared (i.e. when the bit in this register is cleared, the corresponding bit in Interrupt Status Register is always 0).

- Depending on IABG bit setting, uSDHC may be programmed to sample the card interrupt signal during the interrupt period and hold its value in the flip-flop. There will be some delays on the Card Interrupt, asserted from the card, to the time the Host System is informed.
- To detect a CMD line conflict, the Host Driver must set both Command Timeout Error Status Enable and Command CRC Error Status Enable to 1.

Address: Base address + 34h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			DMAESEN	0	TNESEN	0	AC12ESEN	0	DEBESEN	DOESEN	DTOESEN	CIESEN	CEBESEN	COESEN	CTOESEN
W																
Reset	0	0	0	1	0	1	0	1	0	1	1	1	1	1	1	1
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	TPSEN	0	RTESEN	0			CINTSEN	CRMSEN	CINSEN	BRRSEN	BWRSEN	DINTSEN	BGESEN	TCSSEN	CCSEN
W																
Reset	0	1	0	0	0	0	0	1	0	0	1	1	1	1	1	1



**uSDHCx\_INT\_STATUS\_EN field descriptions**

Field	Description
31–29 Reserved	This read-only field is reserved and always has the value 0.
28 DMAESEN	DMA Error Status Enable: 1 Enabled 0 Masked
27 Reserved	This read-only field is reserved and always has the value 0.
26 TNESEN	Tuning Error Status Enable: 1 Enabled 0 Masked
25 Reserved	This read-only field is reserved and always has the value 0.
24 AC12ESEN	Auto CMD12 Error Status Enable: 1 Enabled 0 Masked
23 Reserved	This read-only field is reserved and always has the value 0.
22 DEBESEN	Data End Bit Error Status Enable: 1 Enabled 0 Masked
21 DCESEN	Data CRC Error Status Enable: 1 Enabled 0 Masked
20 DTESEN	Data Timeout Error Status Enable: 1 Enabled 0 Masked
19 CIESEN	Command Index Error Status Enable: 1 Enabled 0 Masked
18 CEBESEN	Command End Bit Error Status Enable: 1 Enabled 0 Masked
17 CCESEN	Command CRC Error Status Enable: 1 Enabled 0 Masked
16 CTESEN	Command Timeout Error Status Enable: 1 Enabled 0 Masked

*Table continues on the next page...*

### uSDHCx\_INT\_STATUS\_EN field descriptions (continued)

Field	Description
15 Reserved	This read-only field is reserved and always has the value 0.
14 TPSEN	Tuning Pass Status Enable  1 Enabled 0 Masked
13 Reserved	This read-only field is reserved and always has the value 0.
12 RTESEN	Re-Tuning Event Status Enable  1 Enabled 0 Masked
11–9 Reserved	This read-only field is reserved and always has the value 0.
8 CINTSEN	Card Interrupt Status Enable:  If this bit is set to 0, the uSDHC will clear the interrupt request to the System. The Card Interrupt detection is stopped when this bit is cleared and restarted when this bit is set to 1. The Host Driver should clear the Card Interrupt Status Enable before servicing the Card Interrupt and should set this bit again after all interrupt requests from the card are cleared to prevent inadvertent interrupts.  1 Enabled 0 Masked
7 CRMSSEN	Card Removal Status Enable:  1 Enabled 0 Masked
6 CINSSSEN	Card Insertion Status Enable:  1 Enabled 0 Masked
5 BRRSEN	Buffer Read Ready Status Enable:  1 Enabled 0 Masked
4 BWRSEN	Buffer Write Ready Status Enable:  1 Enabled 0 Masked
3 DINTSEN	DMA Interrupt Status Enable:  1 Enabled 0 Masked
2 BGESEN	Block Gap Event Status Enable:  1 Enabled 0 Masked
1 TCSEN	Transfer Complete Status Enable:

Table continues on the next page...

**uSDHCx\_INT\_STATUS\_EN field descriptions (continued)**

Field	Description
	1 Enabled 0 Masked
0 CCSEN	Command Complete Status Enable: 1 Enabled 0 Masked

**54.8.16 Interrupt Signal Enable (uSDHCx\_INT\_SIGNAL\_EN)**

This register is used to select which interrupt status is indicated to the Host System as the interrupt. These status bits all share the same interrupt line. Setting any of these bits to 1 enables interrupt generation. The corresponding Status register bit will generate an interrupt when the corresponding interrupt signal enable bit is set.

Address: Base address + 38h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0			DMAEIE	0	TNEIE	0	AC12EIE	0	DEBEIE	DCEIE	DTOEIE	CIEIE	CEBEIE	CCEIE	CTOEIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0	TPIE	0	RTEIE	0			CINTIE	CRMIIE	CINSIE	BRRIIE	BWRIIE	DINTIE	BGEIE	TCIE	CCIE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**uSDHCx\_INT\_SIGNAL\_EN field descriptions**

Field	Description
31–29 Reserved	This read-only field is reserved and always has the value 0.
28 DMAEIE	DMA Error Interrupt Enable: 1 Enable 0 Masked
27 Reserved	This read-only field is reserved and always has the value 0.
26 TNEIE	Tuning Error Interrupt Enable

Table continues on the next page...

### uSDHCx\_INT\_SIGNAL\_EN field descriptions (continued)

Field	Description
	1 Enabled 0 Masked
25 Reserved	This read-only field is reserved and always has the value 0.
24 AC12EIEN	Auto CMD12 Error Interrupt Enable: 1 Enabled 0 Masked
23 Reserved	This read-only field is reserved and always has the value 0.
22 DEBEIEN	Data End Bit Error Interrupt Enable: 1 Enabled 0 Masked
21 DCEIEN	Data CRC Error Interrupt Enable: 1 Enabled 0 Masked
20 DTOEIEN	Data Timeout Error Interrupt Enable: 1 Enabled 0 Masked
19 CIEIEN	Command Index Error Interrupt Enable: 1 Enabled 0 Masked
18 CEBEIEN	Command End Bit Error Interrupt Enable: 1 Enabled 0 Masked
17 CCEIEN	Command CRC Error Interrupt Enable: 1 Enabled 0 Masked
16 CTOEIEN	Command Timeout Error Interrupt Enable 1 Enabled 0 Masked
15 Reserved	This read-only field is reserved and always has the value 0.
14 TPIEN	Tuning Pass Interrupt Enable 1 Enabled 0 Masked
13 Reserved	This read-only field is reserved and always has the value 0.
12 RTEIEN	Re-Tuning Event Interrupt Enable

Table continues on the next page...

**uSDHCx\_INT\_SIGNAL\_EN field descriptions (continued)**

Field	Description
	1 Enabled 0 Masked
11–9 Reserved	This read-only field is reserved and always has the value 0.
8 CINTIEN	Card Interrupt Interrupt Enable: 1 Enabled 0 Masked
7 CRMIEN	Card Removal Interrupt Enable: 1 Enabled 0 Masked
6 CINSIEN	Card Insertion Interrupt Enable: 1 Enabled 0 Masked
5 BRRIEN	Buffer Read Ready Interrupt Enable: 1 Enabled 0 Masked
4 BWRIEN	Buffer Write Ready Interrupt Enable: 1 Enabled 0 Masked
3 DINTIEN	DMA Interrupt Enable: 1 Enabled 0 Masked
2 BGEIEN	Block Gap Event Interrupt Enable: 1 Enabled 0 Masked
1 TCIEN	Transfer Complete Interrupt Enable: 1 Enabled 0 Masked
0 CCIEN	Command Complete Interrupt Enable: 1 Enabled 0 Masked

### 54.8.17 Auto CMD12 Error Status (uSDHCx\_AUTOCMD12\_ERR\_STATUS)

When the Auto CMD12 Error Status bit in the Status register is set, the Host Driver shall check this register to identify what kind of error the Auto CMD12 indicated. This register is valid only when the Auto CMD12 Error status bit is set.

The table below shows the relationship between the Auto CMD12 CRC Error and the Auto CMD12 Command Timeout Error.

**Table 54-71. Relationship Between Command CRC Error and Command Timeout Error for Auto CMD12**

Auto CMD12 CRC Error	Auto CMD12 Timeout Error	Type of Error
0	0	No Error
0	1	Response Timeout Error
1	0	Response CRC Error
1	1	CMD line conflict

Changes in Auto CMD12 Error Status register can be classified in three scenarios:

- When the uSDHC is going to issue an Auto CMD12.
  - Set bit 0 to 1 if the Auto CMD12 can't be issued due to an error in the previous command
  - Set bit 0 to 0 if the Auto CMD12 is issued
- At the end bit of an Auto CMD12 response.
  - Check errors correspond to bits 1-4.
  - Set bits 1-4 corresponding to detected errors.
  - Clear bits 1-4 corresponding to detected errors
- Before reading the Auto CMD12 Error Status bit 7.
  - Set bit 7 to 1 if there is a command that can't be issued
  - Clear bit 7 if there is no command to issue

The timing for generating the Auto CMD12 Error and writing to the Command register are asynchronous. After that, bit 7 shall be sampled when the driver is not writing to the Command register. So it is suggested to read this register only when the AC12E bit in Interrupt Status register is set. An Auto CMD12 Error Interrupt is generated when one of the error bits (0-4) is set to 1. The Command Not Issued By Auto CMD12 Error does not generate an interrupt.

Address: Base address + 3Ch offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0								SMP_CLK_SEL	EXECUTE_TUNING	0					
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								CNIBAC12E	0	AC12IE	AC12CE	AC12EBE	AC12TOE	AC12NE	
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**uSDHCx\_AUTOCMD12\_ERR\_STATUS field descriptions**

Field	Description
31–24 Reserved	This read-only field is reserved and always has the value 0.
23 SMP_CLK_SEL	<p>Sample Clock Select:</p> <p>When std_tuning_en bit is set, this bit is used to select sampling clock to receive CMD and DATA. Otherwise, this bit is reserved. This bit is set by ty tuning procedure and valid after the completion of tuning(When Execute Tuning is cleared). Setting 1 means that tuning is completed successfully and setting 0 means that tuning is failed. Writing 1 to this bit is meaningless and ignored. A tuning circuit is reset by writing to 0. This bit can be cleared with setting Execute Tuning. Once the tuning circuit is reset, it will take time to complete tuning sequence. Therefore, Host Driver should keep this bit to 1 to perform re-tuning sequence to complete re-tuning sequence in a short time. Change of this bit is not allowed while the Host controller us receiving response or a read data block.</p>

*Table continues on the next page...*

**uSDHCx\_AUTOCMD12\_ERR\_STATUS field descriptions (continued)**

Field	Description
	1 Tuned clock is used to sample data 0 Fixed clock is used to sample data
22 EXECUTE_TUNING	Execute Tuning: When std_tuning_en bit is set, this bit is used to start tuning procedure. Otherwise, this bit is reserved. This bit is set to start tuning procedure and automatically cleared when runing procedure is completed. The result of tuning is indicated to sam_clk_sel bit. Tuning procedure is aborted by writing 0.
21–8 Reserved	This read-only field is reserved and always has the value 0.
7 CNIBAC12E	Command Not Issued By Auto CMD12 Error: Setting this bit to 1 means CMD_wo_DAT is not executed due to an Auto CMD12 Error (D04-D01) in this register. 1 Not Issued 0 No error
6–5 Reserved	This read-only field is reserved and always has the value 0.
4 AC12IE	Auto CMD12 Index Error: Occurs if the Command Index error occurs in response to a command. 1 Error, the CMD index in response is not CMD12 0 No error
3 AC12CE	Auto CMD12 CRC Error: Occurs when detecting a CRC error in the command response. 1 CRC Error Met in Auto CMD12 Response 0 No CRC error
2 AC12EBE	Auto CMD12 End Bit Error: Occurs when detecting that the end bit of command response is 0 which should be 1. 1 End Bit Error Generated 0 No error
1 AC12TOE	Auto CMD12 Timeout Error: Occurs if no response is returned within 64 SDCLK cycles from the end bit of the command. If this bit is set to 1, the other error status bits (2-4) have no meaning. 1 Time out 0 No error
0 AC12NE	Auto CMD12 Not Executed: If memory multiple block data transfer is not started, due to a command error, this bit is not set because it is not necessary to issue an Auto CMD12. Setting this bit to 1 means the uSDHC cannot issue the Auto CMD12 to stop a memory multiple block data transfer due to some error. If this bit is set to 1, other error status bits (1-4) have no meaning. 1 Not executed 0 Executed

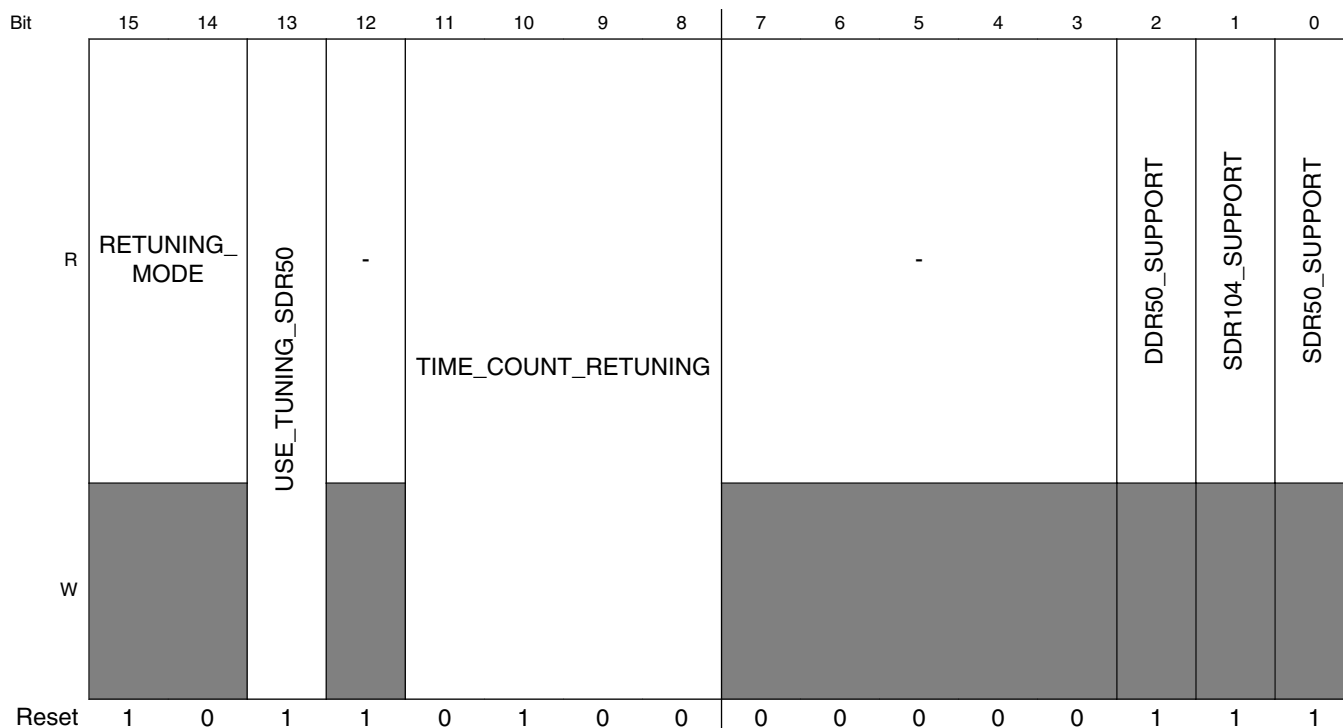


### 54.8.18 Host Controller Capabilities (uSDHCx\_HOST\_CTRL\_CAP)

This register provides the Host Driver with information specific to the uSDHC implementation. The value in this register is the power-on-reset value, and does not change with a software reset. Any write to this register is ignored.

Address: Base address + 40h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0					VS18	VS30	VS33	SRS	DMAS	HSS	ADMAS	0	MBL[2:0]		
W																
Reset	0	0	0	0	0	1	1	1	1	1	1	1	0	0	1	1


**uSDHCx\_HOST\_CTRL\_CAP field descriptions**

Field	Description
31–27 Reserved	This read-only field is reserved and always has the value 0.
26 VS18	Voltage Support 1.8V: This bit shall depend on the Host System ability.  1 1.8V supported 0 1.8V not supported
25 VS30	Voltage Support 3.0V: This bit shall depend on the Host System ability.  1 3.0V supported 0 3.0V not supported
24 VS33	Voltage Support 3.3V: This bit shall depend on the Host System ability.  1 3.3V supported 0 3.3V not supported
23 SRS	Suspend / Resume Support: This bit indicates whether the uSDHC supports Suspend / Resume functionality. If this bit is 0, the Suspend and Resume mechanism, as well as the Read Wait, are not supported, and the Host Driver shall not issue either Suspend or Resume commands.  1 Supported 0 Not supported

Table continues on the next page...

**uSDHCx\_HOST\_CTRL\_CAP field descriptions (continued)**

Field	Description
22 DMAS	<p>DMA Support:</p> <p>This bit indicates whether the uSDHC is capable of using the internal DMA to transfer data between system memory and the data buffer directly.</p> <p>1 DMA Supported 0 DMA not supported</p>
21 HSS	<p>High Speed Support:</p> <p>This bit indicates whether the uSDHC supports High Speed mode and the Host System can supply a SD Clock frequency from 25 MHz to 50 MHz.</p> <p>1 High Speed Supported 0 High Speed Not Supported</p>
20 ADMAS	<p>ADMA Support:</p> <p>This bit indicates whether the uSDHC supports the ADMA feature.</p> <p>1 Advanced DMA Supported 0 Advanced DMA Not supported</p>
19 Reserved	This read-only field is reserved and always has the value 0.
18–16 MBL[2:0]	<p>Max Block Length:</p> <p>This value indicates the maximum block size that the Host Driver can read and write to the buffer in the uSDHC. The buffer shall transfer block size without wait cycles.</p> <p>000 512 bytes 001 1024 bytes 010 2048 bytes 011 4096 bytes</p>
15–14 RETUNING_ MODE	<p>Retuning Mode:</p> <p>This bit selects retuning method.</p> <p>00 Mode 1 01 Mode 2 10 Mode 3 11 Reserved</p>
13 USE_TUNING_ SDR50	<p>Use Tuning for SDR50:</p> <p>This bit is set to 1. Host controller requires tuning to operate SDR50</p> <p>1 SDR50 requires tuning 0 SDR does not require tuning</p>
12 -	Reserved
11–8 TIME_COUNT_ RETUNING	<p>Time counter for retuning:</p> <p>This bit indicates an initial value of the Retuning Timer for Re-Tuning Mode1 and 3. Setting to 0 disables Retuning Timer.</p>
7–3 -	

*Table continues on the next page...*

### uSDHCx\_HOST\_CTRL\_CAP field descriptions (continued)

Field	Description
2 DDR50_SUPPORT	DDR50 support: This bit indicates support of DDR50 mode.
1 SDR104_SUPPORT	SDR104 support: This bit indicates support of SDR104 mode.
0 SDR50_SUPPORT	SDR50 support: This bit indicates support of SDR50 mode.

## 54.8.19 Watermark Level (uSDHCx\_WTMK\_LVL)

Both write and read watermark levels (FIFO threshold) are configurable. Their value can range from 1 to 128 words. Both write and read burst lengths are also Configurable. Their value can range from 1 to 31 words.

Address: Base address + 44h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0			WR_BRST_				WR_WML[7:0]									0			RD_BRST_				RD_WML[7:0]								
W				LEN[4:0]																LEN[4:0]												
Reset	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0

### uSDHCx\_WTMK\_LVL field descriptions

Field	Description
31–29 Reserved	This read-only field is reserved and always has the value 0.
28–24 WR_BRST_LEN[4:0]	Write Burst Length: The number of words the uSDHC writes in a single burst. The write burst length must be less than or equal to the write watermark level, and all bursts within a watermark level transfer will be in back-to-back mode. On reset, this field will be 8. Writing 0 to this field will result in '01000' (i.e. it is not able to clear this field).
23–16 WR_WML[7:0]	Write Watermark Level: The number of words used as the watermark level (FIFO threshold) in a DMA write operation. Also the number of words as a sequence of write bursts in back-to-back mode. The maximum legal value for the write watermark level is 128.
15–13 Reserved	This read-only field is reserved and always has the value 0.
12–8 RD_BRST_LEN[4:0]	Read Burst Length: The number of words the uSDHC reads in a single burst. The read burst length must be less than or equal to the read watermark level, and all bursts within a watermark level transfer will be in back-to-back mode. On reset, this field will be 8. Writing 0 to this field will result in '01000' (i.e. it is not able to clear this field).

Table continues on the next page...

**uSDHCx\_WTMK\_LVL field descriptions (continued)**

Field	Description
7–0 RD_WML[7:0]	Read Watermark Level: The number of words used as the watermark level (FIFO threshold) in a DMA read operation. Also the number of words as a sequence of read bursts in back-to-back mode. The maximum legal value for the read water mark level is 128.

1. Due to system restriction, the actual burst length may not exceed 16.
2. Due to system restriction, the actual burst length may not exceed 16.

**54.8.20 Mixer Control (uSDHCx\_MIX\_CTRL)**

This register is used to DMA and data transfer. To prevent data loss, The software should check if data transfer is active before writing to this register. These bits are DPSEL, MBSEL, DTDSEL, AC12EN, BCEN and DMAEN.

**Table 54-75. Transfer Type Register Setting for Various Transfer Types**

Multi/Single Block Select	Block Count Enable	Block Count	Function
0	Don't Care	Don't Care	Single Transfer
1	0	Don't Care	Infinite Transfer
1	1	Positive Number	Multiple Transfer
1	1	Zero	No Data Transfer

Address: Base address + 48h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R				0										0		
W	-	-	-				FBCLK_SEL	AUTO_TUNE_EN	SMP_CLK_SEL	EXE_TUNE						
Reset	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R				0												
W									AC23EN	NIBBLE_POS	MSBSEL	DTDSEL	DDR_EN	AC12EN	BCEN	DMAEN
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### uSDHCx\_MIX\_CTRL field descriptions

Field	Description
31 -	Reserved. Always write as 1
30 -	Reserved. Always write as 0.
29 -	Reserved. Always write as 0.
28–26 Reserved	This read-only field is reserved and always has the value 0.
25 FBCLK_SEL	Feedback clock source selection (Only used for SD3.0, SDR104 mode)  1 feedback clock comes from the ipp_card_clk_out 0 feedback clock comes from the loopback CLK
24 AUTO_TUNE_EN	Auto tuning enable (Only used for SD3.0, SDR104 mode)  1 enable auto tuning 0 disable auto tuning
23 SMP_CLK_SEL	When std_tuning_en is 0, this bit is used to select Tuned clock or Fixed clock to sample data/cmd (Only used for SD3.0, SDR104 mode)  1 Tuned clock is used to sample data/cmd 0 Fixed clock is used to sample data/cmd
22 EXE_TUNE	Execute Tuning: (Only used for SD3.0, SDR104 mode)  When std_tuning_en is 0, this bit is set to 1 to indicate the Host Driver is starting tuning procedure. Tuning procedure is aborted by writing 0.  1 Execute Tuning 0 Not Tuned or Tuning Completed
21–8 Reserved	This read-only field is reserved and always has the value 0.
7 AC23EN	Auto CMD23 Enable  When this bit is set to 1, the Host Controller issues a CMD23 automatically before issuing a command specified in the Command Register.
6 NIBBLE_POS	In DDR 4-bit mode nibble position indication. 0- the sequence is 'odd high nibble -> even high nibble -> odd low nibble -> even low nibble'; 1- the sequence is 'odd high nibble -> odd low nibble -> even high nibble -> even low nibble'.
5 MSBSEL	Multi / Single Block Select:  This bit enables multiple block DATA line data transfers. For any other commands, this bit shall be set to 0. If this bit is 0, it is not necessary to set the Block Count register. (Refer to <a href="#">Command Transfer Type (uSDHC_CMD_XFR_TYP)</a> ).  1 Multiple Blocks 0 Single Block
4 DTDSEL	Data Transfer Direction Select:  This bit defines the direction of DATA line data transfers. The bit is set to 1 by the Host Driver to transfer data from the SD card to the uSDHC and is set to 0 for all other commands.  1 Read (Card to Host) 0 Write (Host to Card)

Table continues on the next page...

**uSDHCx\_MIX\_CTRL field descriptions (continued)**

Field	Description
3 DDR_EN	Dual Data Rate mode selection
2 AC12EN	<p>Auto CMD12 Enable:</p> <p>Multiple block transfers for memory require a CMD12 to stop the transaction. When this bit is set to 1, the uSDHC will issue a CMD12 automatically when the last block transfer has completed. The Host Driver shall not set this bit to issue commands that do not require CMD12 to stop a multiple block data transfer. In particular, secure commands defined in File Security Specification (see reference list) do not require CMD12. In single block transfer, the uSDHC will ignore this bit no matter if it is set or not.</p> <p>1 Enable 0 Disable</p>
1 BCEN	<p>Block Count Enable:</p> <p>This bit is used to enable the Block Count register, which is only relevant for multiple block transfers. When this bit is 0, the internal counter for block is disabled, which is useful in executing an infinite transfer.</p> <p>1 Enable 0 Disable</p>
0 DMAEN	<p>DMA Enable:</p> <p>This bit enables DMA functionality. If this bit is set to 1, a DMA operation shall begin when the Host Driver sets the DPSEL bit of this register. Whether the Simple DMA, or the Advanced DMA, is active depends on the DMA Select field of the Protocol Control register.</p> <p>1 Enable 0 Disable</p>

**54.8.21 Force Event (uSDHCx\_FORCE\_EVENT)**

The Force Event Register is not a physically implemented register. Rather, it is an address at which the Interrupt Status Register can be written if the corresponding bit of the Interrupt Status Enable Register is set. This register is a write only register and writing 0 to it has no effect. Writing 1 to this register actually sets the corresponding bit of Interrupt Status Register. A read from this register always results in 0's. In order to change the corresponding status bits in the Interrupt Status Register, make sure to set IPGEN bit in System Control Register so that ipg\_clk is always active.

Forcing a card interrupt will generate a short pulse on the DATA1 line, and the driver may treat this interrupt as a normal interrupt. The interrupt service routine may skip polling the card interrupt factor as the interrupt is self cleared.

## uSDHC Memory Map/Register Definition

Address: Base address + 50h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
W	FEVTCINT			FEVTDMAE		FEVTTNE		FEVTAC12E		FEVTDEBE	FEVTDCE	FEVTDTOE	FEVTCIE	FEVTCBE	FEVTCCE	FEVTCIOE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R				0					0	0		0	0	0	0	0
W									FEVTCNIBAC12E			FEVTAC12IE	FEVTAC12EBE	FEVTAC12CE	FEVTAC12TOE	FEVTAC12NE
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

### uSDHCx\_FORCE\_EVENT field descriptions

Field	Description
31 FEVTCINT	Force Event Card Interrupt: Writing 1 to this bit generates a short low-level pulse on the internal DATA1 line, as if a self clearing interrupt was received from the external card. If enabled, the CINT bit will be set and the interrupt service routine may treat this interrupt as a normal interrupt from the external card.
30–29 Reserved	This read-only field is reserved and always has the value 0.
28 FEVTDMAE	Force Event DMA Error: Forces the DMAE bit of Interrupt Status Register to be set
27 Reserved	This read-only field is reserved and always has the value 0.
26 FEVTTNE	Force Tuning Error: Forces the TNE bit of Interrupt Status Register to be set
25 Reserved	This read-only field is reserved and always has the value 0.

Table continues on the next page...



**uSDHCx\_FORCE\_EVENT field descriptions (continued)**

Field	Description
24 FEVTAC12E	Force Event Auto Command 12 Error: Forces the AC12E bit of Interrupt Status Register to be set
23 Reserved	This read-only field is reserved and always has the value 0.
22 FEVTDEBE	Force Event Data End Bit Error: Forces the DEBE bit of Interrupt Status Register to be set
21 FEVTDCE	Force Event Data CRC Error: Forces the DCE bit of Interrupt Status Register to be set
20 FEVTDTOE	Force Event Data Time Out Error: Force the DTOE bit of Interrupt Status Register to be set
19 FEVTCIE	Force Event Command Index Error: Forces the CCE bit of Interrupt Status Register to be set
18 FEVTCBE	Force Event Command End Bit Error: Forces the CEBE bit of Interrupt Status Register to be set
17 FEVTCCE	Force Event Command CRC Error: Forces the CCE bit of Interrupt Status Register to be set
16 FEVCTOE	Force Event Command Time Out Error: Forces the CTOE bit of Interrupt Status Register to be set
15–8 Reserved	This read-only field is reserved and always has the value 0.
7 FEVTCNIBAC12E	Force Event Command Not Executed By Auto Command 12 Error: Forces the CNIBAC12E bit in the Auto Command12 Error Status Register to be set
6–5 Reserved	This read-only field is reserved and always has the value 0.
4 FEVTAC12IE	Force Event Auto Command 12 Index Error: Forces the AC12IE bit in the Auto Command12 Error Status Register to be set
3 FEVTAC12EBE	Force Event Auto Command 12 End Bit Error: Forces the AC12EBE bit in the Auto Command12 Error Status Register to be set
2 FEVTAC12CE	Force Event Auto Command 12 CRC Error: Forces the AC12CE bit in the Auto Command12 Error Status Register to be set
1 FEVTAC12TOE	Force Event Auto Command 12 Time Out Error: Forces the AC12TOE bit in the Auto Command12 Error Status Register to be set
0 FEVTAC12NE	Force Event Auto Command 12 Not Executed: Forces the AC12NE bit in the Auto Command12 Error Status Register to be set

### 54.8.22 ADMA Error Status Register (uSDHCx\_ADMA\_ERR\_STATUS)

When an ADMA Error Interrupt has occurred, the ADMA Error States field in this register holds the ADMA state and the ADMA System Address register holds the address around the error descriptor.

For recovering from this error, the Host Driver requires the ADMA state to identify the error descriptor address as follows:

- **ST\_STOP:** Previous location set in the ADMA System Address register is the error descriptor address
- **ST\_FDS:** Current location set in the ADMA System Address register is the error descriptor address
- **ST\_CADR:** This state is never set because it only increments the descriptor pointer and doesn't generate an ADMA error
- **ST\_TFR:** Previous location set in the ADMA System Address register is the error descriptor address

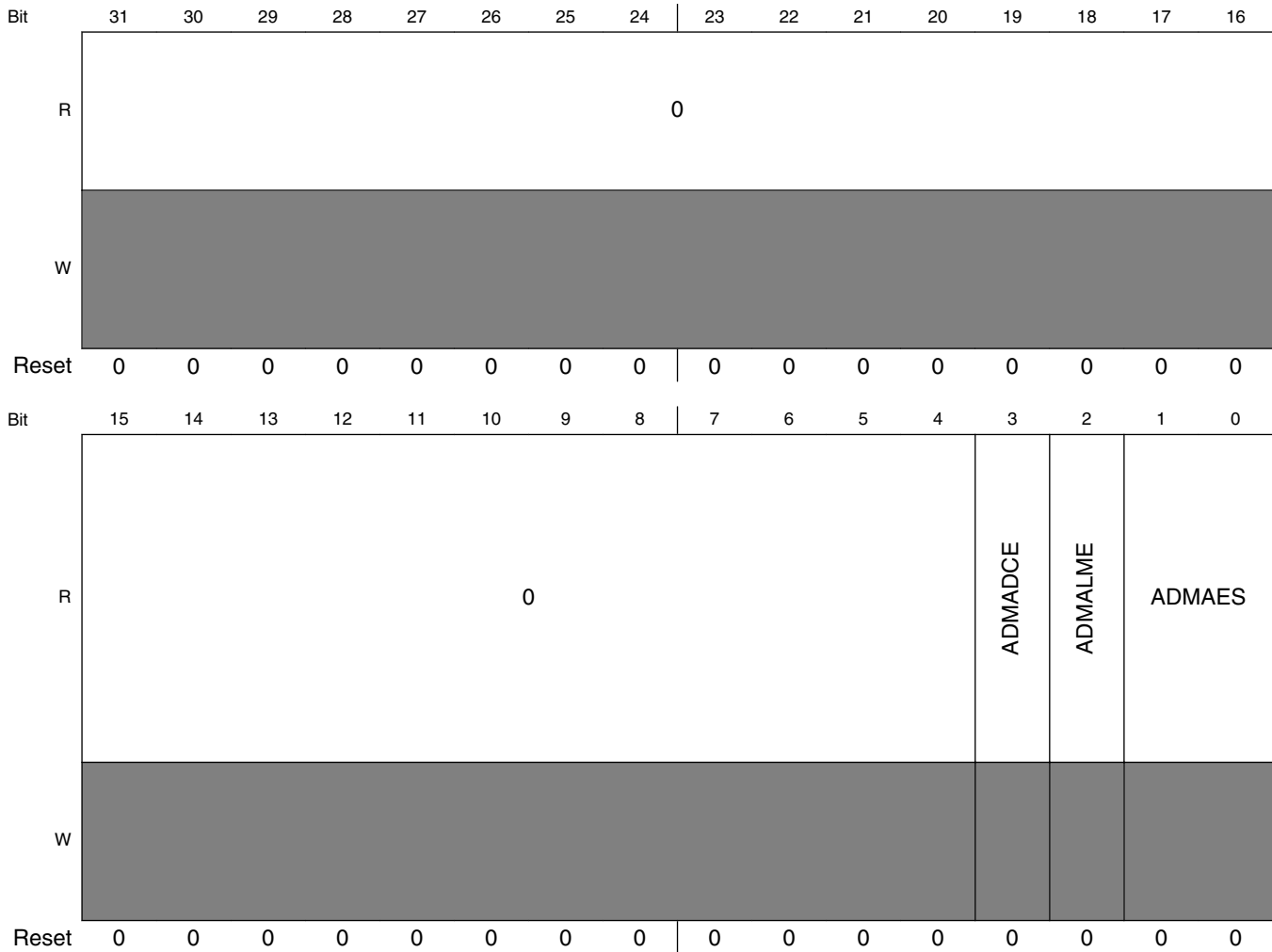
In case of a write operation, the Host Driver should use the ACMD22 to get the number of the written block, rather than using this information, since unwritten data may exist in the Host Controller.

The Host Controller generates the ADMA Error Interrupt when it detects invalid descriptor data (Valid=0) in the ST\_FDS state. The Host Driver can distinguish this error by reading the Valid bit of the error descriptor.

**Table 54-78. ADMA Error State Coding**

D01-D00	ADMA Error State (when error has occurred)	Contents of ADMA System Address Register
00	ST_STOP (Stop DMA)	Holds the address of the next executable Descriptor command
01	ST_FDS (Fetch Descriptor)	Holds the valid Descriptor address
10	ST_CADR (Change Address)	No ADMA Error is generated
11	ST_TFR (Transfer Data)	Holds the address of the next executable Descriptor command

Address: Base address + 54h offset

**uSDHCx\_ADMA\_ERR\_STATUS field descriptions**

Field	Description
31–4 Reserved	This read-only field is reserved and always has the value 0.
3 ADMADCE	ADMA Descriptor Error: This error occurs when invalid descriptor fetched by ADMA:  1 Error 0 No Error
2 ADMALME	ADMA Length Mismatch Error: This error occurs in the following 2 cases: <ul style="list-style-type: none"> <li>While the Block Count Enable is being set, the total data length specified by the Descriptor table is different from that specified by the Block Count and Block Length</li> <li>Total data length can not be divided by the block length</li> </ul> 1 Error 0 No Error

*Table continues on the next page...*

**uSDHCx\_ADMA\_ERR\_STATUS field descriptions (continued)**

Field	Description
1–0 ADMAES	ADMA Error State (when ADMA Error is occurred.):  This field indicates the state of the ADMA when an error has occurred during an ADMA data transfer. Refer to <a href="#">ADMA Error Status Register (uSDHCx_ADMA_ERR_STATUS)</a> for more details.

**54.8.23 ADMA System Address (uSDHCx\_ADMA\_SYS\_ADDR)**

This register contains the physical system memory address used for ADMA transfers.

Address: Base address + 58h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R																																0
W																																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**uSDHCx\_ADMA\_SYS\_ADDR field descriptions**

Field	Description
31–2 ADS_ ADDR[31:0]	ADMA System Address:  This register holds the word address of the executing command in the Descriptor table. At the start of ADMA, the Host Driver shall set the start address of the Descriptor table. The ADMA engine increments this register address whenever fetching a Descriptor command. When the ADMA is stopped at the Block Gap, this register indicates the address of the next executable Descriptor command. When the ADMA Error Interrupt is generated, this register shall hold the valid Descriptor address depending on the ADMA state. The lower 2 bits of this register is tied to '0' so the ADMA address is always word aligned.  Since this register supports dynamic address reflecting, when TC bit is set, it automatically alters the value of internal address counter, so SW cannot change this register when TC bit is set. Such restriction is also listed in <a href="#">Software Restrictions</a> .
1–0 Reserved	This read-only field is reserved and always has the value 0.

## 54.8.24 DLL (Delay Line) Control (uSDHCx\_DLL\_CTRL)

This register contains control bits for DLL.

Address: Base address + 60h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	DLL_CTRL_REF_UPDATE_INT[3:0]				DLL_CTRL_SLV_UPDATE_INT[7:0]								0	DLL_CTRL_SLV_DLY_TARGET1		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	DLL_CTRL_SLV_OVERRIDE_VAL[6:0]							DLL_CTRL_SLV_OVERRIDE	DLL_CTRL_GATE_UPDATE	DLL_CTRL_SLV_DLY_TARGET0				DLL_CTRL_SLV_FORCE_UPD	DLL_CTRL_RESET	DLL_CTRL_ENABLE
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**uSDHCx\_DLL\_CTRL field descriptions**

Field	Description
31–28 DLL_CTRL_REF_UPDATE_INT[3:0]	DLL control loop update interval. The interval cycle is $(2 + \text{REF\_UPDATE\_INT}) * \text{ref\_clock}$ . By default, the DLL control loop shall update every two ref_clock cycles. It should be noted that increasing the reference delay-line update interval reduces the ability of the DLL to adjust to fast changes in conditions that may effect the delay (such as voltage and temperature)
27–20 DLL_CTRL_SLV_UPDATE_INT[7:0]	Slave delay line update interval. If default 0 is used, it means 256 cycles of ref_clock. A value of 0x0f results in 15 cycles and so on. Note that software can always cause an update of the slave-delay line using the SLV_FORCE_UPDATE register. Note that the slave delay line will also update automatically when the reference DLL transitions to a locked state (from an un-locked state).
19 Reserved	This read-only field is reserved and always has the value 0.
18–16 DLL_CTRL_SLV_DLY_TARGET1	Refer to DLL_CTRL_SLV_DLY_TARGET0 below.
15–9 DLL_CTRL_SLV_OVERRIDE_VAL[6:0]	When SLV_OVERRIDE=1 This field is used to select 1 of 128 physical taps manually. A value of 0 selects tap 1, and a value of 0x7f selects tap 128.

Table continues on the next page...

### uSDHCx\_DLL\_CTRL field descriptions (continued)

Field	Description
8 DLL_CTRL_ SLV_OVERRIDE	Set this bit to 1 to Enable manual override for slave delay chain using SLV_OVERRIDE_VAL; to set 0 to disable manual override. This feature does not require the DLL to be enabled using the ENABLE bit. In fact to reduce power, if SLV_OVERRIDE is used, it is recommended to disable the DLL with ENABLE=0
7 DLL_CTRL_ GATE_UPDATE	Set this bit to 1 to prevent the DLL from updating (since when clock_in exists, glitches may appear during DLL updates). This bit may be used by software if such a condition occurs. Clear the bit to 0 to allow the DLL to update automatically.
6-3 DLL_CTRL_ SLV_DLY_ TARGET0	The delay target for the uSDHC loopback read clock can be programmed in 1/16th increments of an ref_clock half-period. The delay is $((DLL\_CTRL\_SLV\_DLY\_TARGET1 + 1) * ref\_clock/2)/16$ So the input read-clock can be delayed relative input data from $(ref\_clock/2)/16$ to $ref\_clock*4$
2 DLL_CTRL_ SLV_FORCE_ UPD	Setting this bit to 1, forces the slave delay line to update to the DLL calibrated value immediately. The slave delay line shall update automatically based on the SLV_UPDATE_INT interval or when a DLL lock condition is sensed. Subsequent forcing of the slave-line update can only occur if SLV_FORCE_UP is set back to 0 and then asserted again (edge triggered). Be sure to use it when uSDHC is idle. This function may not work when uSDHC is working on data/cmd/response.
1 DLL_CTRL_ RESET	Setting this bit to 1 force a reset on DLL. This will cause the DLL to lose lock and re-calibrate to detect an ref_clock half period phase shift. This signal is used by the DLL as edge-sensitive, so in order to create a subsequent reset, RESET must be taken low and then asserted again
0 DLL_CTRL_ ENABLE	Set this bit to 1 to enable the DLL and delay chain; otherwise; set to 0 to bypasses DLL. Note that using the slave delay line override feature with SLV_OVERRIDE and SLV_OVERRIDE_VAL, the DLL does not need to be enabled

### 54.8.25 DLL Status (uSDHCx\_DLL\_STATUS)

This register contains the DLL status information. All bits are read only and will read the same as the power-reset value.

Address: Base address + 64h offset

[illegible]

**uSDHCx\_DLL\_STATUS field descriptions**

Field	Description
31–16 Reserved	This read-only field is reserved and always has the value 0.
15–9 DLL_STS_REF_SEL[6:0]	Reference delay line select taps. This is encoded by 7 bits for 127 taps.
8–2 DLL_STS_SLV_SEL[6:0]	Slave delay line select status. This is the instant value generated from reference chain. Since the reference chain can only be updated when ref_clock is detected, this value should be the right value to be updated when the reference is locked.
1 DLL_STS_REF_LOCK	Reference DLL lock status. This signifies that the DLL has detected and locked to a half-phase ref_clock shift, allowing the slave delay-line to perform programmed clock delays
0 DLL_STS_SLV_LOCK	Slave delay-line lock status. This signifies that a valid calibration has been set to the slave-delay line and that the slave-delay line is implementing the programmed delay value

## 54.8.26 CLK Tuning Control and Status (uSDHCx\_CLK\_TUNE\_CTRL\_STATUS)

This register contains the Clock Tuning Control status information. All bits are read only and will read the same as the power-reset value. This register is added to support SD3.0 UHS-I SDR104 mode.

Address: Base address + 68h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	PRE_ERR	TAP_SEL_PRE[6:0]							TAP_SEL_OUT[3:0]				TAP_SEL_POST[3:0]			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	NXT_ERR	DLY_CELL_SET_PRE[6:0]							DLY_CELL_SET_OUT[6:0]				DLY_CELL_SET_POST[6:0]			
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**uSDHCx\_CLK\_TUNE\_CTRL\_STATUS field descriptions**

Field	Description
31 PRE_ERR	PRE error which means the number of delay cells added on the feedback clock is too small. It's valid only when SMP_CLK_SEL of Mix control register(bit23 of 0x48) is enabled.

*Table continues on the next page...*



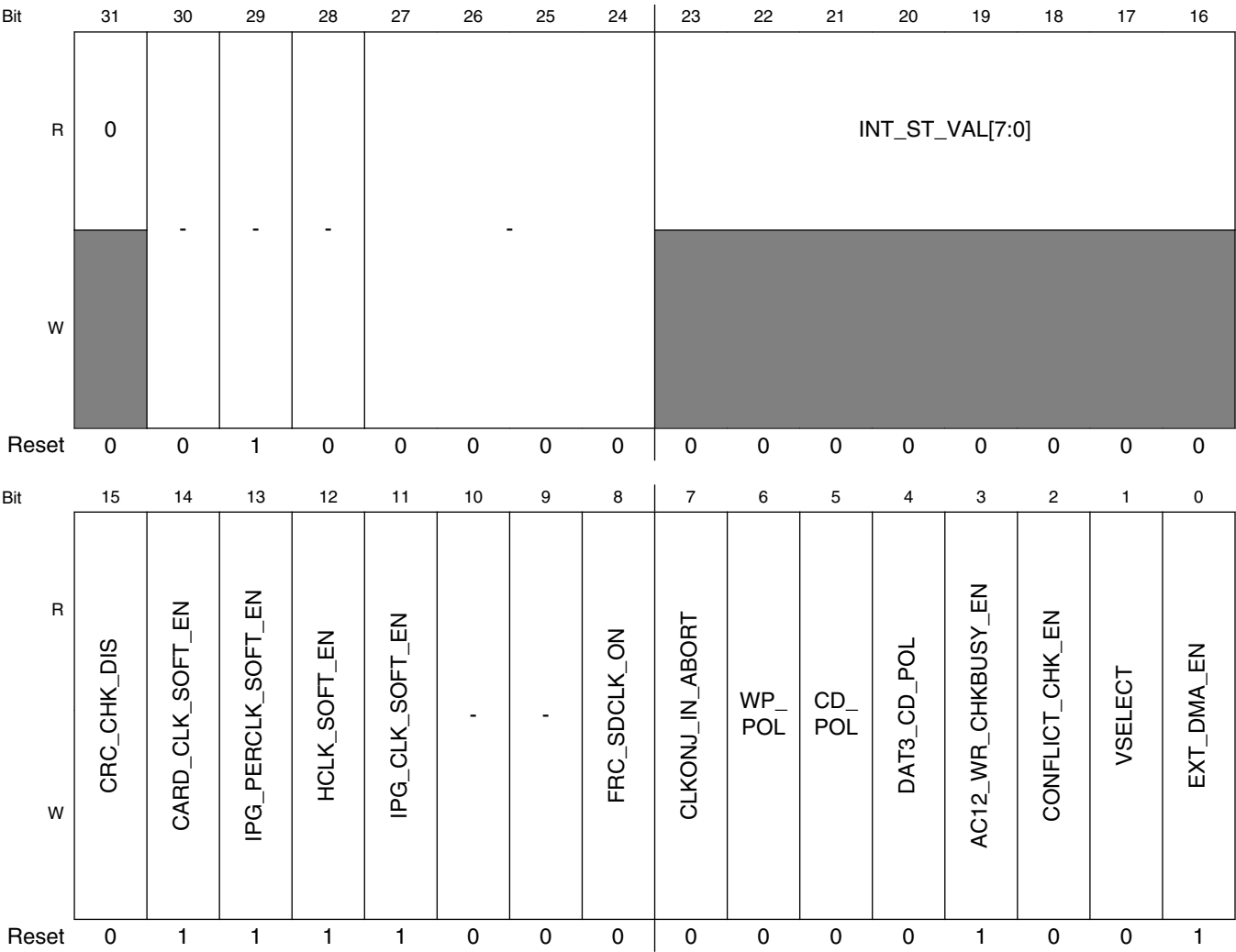
**uSDHCx\_CLK\_TUNE\_CTRL\_STATUS field descriptions (continued)**

Field	Description
30–24 TAP_SEL_ PRE[6:0]	<p>Reflects the number of delay cells added on the feedback clock between the feedback clock and CLK_PRE.</p> <p>When AUTO_TUNE_EN(bit24 of 0x48) is disabled, TAP_SEL_PRE is always equal to DLY_CELL_SET_PRE.</p> <p>When AUTO_TUNE_EN(bit24 of 0x48) is enabled, TAP_SEL_PRE will be updated automatically according to the status of the auto tuning circuit to adjust the sample clock phase.</p>
23–20 TAP_SEL_ OUT[3:0]	Reflect the number of delay cells added on the feedback clock between CLK_PRE and CLK_OUT.
19–16 TAP_SEL_ POST[3:0]	Reflect the number of delay cells added on the feedback clock between CLK_OUT and CLK_POST.
15 NXT_ERR	NXT error which means the number of delay cells added on the feedback clock is too large. It's valid only when SMP_CLK_SEL of Mix control register(bit23 of 0x48) is enabled.
14–8 DLY_CELL_ SET_PRE[6:0]	Set the number of delay cells on the feedback clock between the feedback clock and CLK_PRE.
7–4 DLY_CELL_ SET_OUT[6:0]	Set the number of delay cells on the feedback clock between CLK_PRE and CLK_OUT.
3–0 DLY_CELL_ SET_POST[6:0]	Set the number of delay cells on the feedback clock between CLK_OUT and CLK_POST.

### 54.8.27 Vendor Specific Register (uSDHCx\_VEND\_SPEC)

This register contains the vendor specific control/status register.

Address: Base address + C0h offset



uSDHCx\_VEND\_SPEC field descriptions

Field	Description
31 Reserved	This read-only field is reserved and always has the value 0.
30 -	Reserved. Always write as 0.
29 -	Reserved. Always write as 1.

Table continues on the next page...

**uSDHCx\_VEND\_SPEC field descriptions (continued)**

Field	Description
28 -	Reserved. Always write as 0.
27–24 -	Reserved. Always write as 4'b0000.
23–16 INT_ST_VAL[7:0]	Internal State Value Internal state value, reflecting the corresponding state value selected by Debug Select field. This field is read-only and write to this field does not have effect.
15 CRC_CHK_DIS	CRC check disable 0 check CRC16 for every read data packet and check CRC bits for every write data packet 1 ignore CRC16 check for every read data packet and ignore CRC bits check for every write data packet
14 CARD_CLK_SOFT_EN	card clock software enable 0 gate off the sd_clk 1 enable the sd_clk
13 IPG_PERCLK_SOFT_EN	ipg_perclk software enable 0 gate off the ipg_perclk 1 enable the ipg_perclk
12 HCLK_SOFT_EN	Please note, hardware auto-enables the AHB clock when the internal DMA is enabled even if HCLK_SOFT_EN is 0. AHB clock software enable 0 gate off the AHB clock. 1 enable the AHB clock.
11 IPG_CLK_SOFT_EN	IPG_CLK software enable 0 gate off the IPG_CLK 1 enable the IPG_CLK
10 -	Reserved. Always write as 0.
9 -	Reserved. Always write as 0.
8 FRC_SDCLK_ON	Force CLK output active: 0 CLK active or inactive is fully controlled by the hardware 1 force CLK active
7 CLKONJ_IN_ABORT	Only for debug. Force CLK output active when sending Abort command: 0 the CLK output is active when sending abort command while data is transmitting even if the internal FIFO is full(for read) or empty(for write) 1 the CLK output is inactive when sending abort command while data is transmitting if the internal FIFO is full(for read) or empty(for write)
6 WP_POL	Only for debug. Polarity of the WP pin:

*Table continues on the next page...*

### uSDHCx\_VEND\_SPEC field descriptions (continued)

Field	Description
	0 WP pin is high active 1 WP pin is low active
5 CD_POL	Only for debug. Polarity of the CD_B pin: 0 CD_B pin is low active 1 CD_B pin is high active
4 DAT3_CD_POL	Only for debug. Polarity of DATA3 pin when it's used as card detection: 0 card detected when DATA3 is high 1 card detected when DATA3 is low
3 AC12_WR_CHKBUSY_EN	Check busy enable after auto CMD12 for write data packet 0 Do not check busy after auto CMD12 for write data packet 1 Check busy after auto CMD12 for write data packet
2 CONFLICT_CHK_EN	It's not implemented in uSDHC IP. Conflict check enable. 0 conflict check disable 1 conflict check enable
1 VSELECT	Voltage Selection Change the value of output signal VSELECT, to control the voltage on pads for external card. There must be a control circuit out of uSDHC to change the voltage on pads. 1 Change the voltage to low voltage range, around 1.8V 0 Change the voltage to high voltage range, around 3.0V
0 EXT_DMA_EN	External DMA Request Enable Enable the request to external DMA. When the internal DMA (either Simple DMA or Advanced DMA) is not in use, and this bit is set, uSDHC will send out DMA request when the internal buffer is ready. This bit is particularly useful when transferring data by ARM platform polling mode, and it is not allowed to send out the external DMA request. By default, this bit is set. 0 In any scenario, uSDHC does not send out external DMA request 1 When internal DMA is not active, the external DMA request will be sent out

## 54.8.28 MMC Boot Register (uSDHCx\_MMC\_BOOT)

This register contains the MMC Fast Boot control register.

Address: Base address + C4h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	BOOT_BLK_CNT[15:0]															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								DISABLE_TIME_OUT	AUTO_SABG_EN	BOOT_EN	BOOT_MODE	BOOT_ACK	DTCV_ACK[3:0]		
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**uSDHCx\_MMC\_BOOT field descriptions**

Field	Description
31–16 BOOT_BLK_CNT[15:0]	The value defines the Stop At Block Gap value of automatic mode. When received card block cnt is equal to (BLK_CNT - BOOT_BLK_CNT) and AUTO_SABG_EN is 1, then Stop At Block Gap. Here, BLK_CNT is defined in the Block Attributes Register, bit31-16 of 0x04.
15–9 Reserved	This read-only field is reserved and always has the value 0.
8 DISABLE_TIME_OUT	Please note, when this bit is set, there is no timeout check no matter whether boot_en is set or not. Disable time out. 0 Enable time out 1 Disable time out
7 AUTO_SABG_EN	During boot, enable auto stop at block gap function. This function will be triggered, and host will stop at block gap when received card block cnt is equal to (BLK_CNT - BOOT_BLK_CNT).
6 BOOT_EN	Boot mode enable. 0 Fast boot disable 1 Fast boot enable
5 BOOT_MODE	Boot mode select. 0 Normal boot 1 Alternative boot

Table continues on the next page...

## uSDHCx\_MMC\_BOOT field descriptions (continued)

Field	Description
4 BOOT_ACK	Boot ack mode select. 0 No ack 1 Ack
3–0 DTCV_ ACK[3:0]	Boot ACK time out counter value. 0000 SDCLK x 2 <sup>13</sup> 0001 SDCLK x 2 <sup>14</sup> 0010 SDCLK x 2 <sup>15</sup> 0011 SDCLK x 2 <sup>16</sup> 0100 SDCLK x 2 <sup>17</sup> 0101 SDCLK x 2 <sup>18</sup> 0110 SDCLK x 2 <sup>19</sup> 0111 SDCLK x 2 <sup>20</sup> 1110 SDCLK x 2 <sup>27</sup> 1111 SDCLK x 2 <sup>28</sup>

## 54.8.29 Vendor Specific 2 Register (uSDHCx\_VEND\_SPEC2)

This register contains the vendor specific control 2 register.

Address: Base address + C8h offset

Bit	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
R	0															
W																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	0								CARD_INT_AUTO_	TUNING_CMD_EN	TUNING_1bit_EN	TUNING_8bit_EN	CARD_INT_D3_	SDR104_NSD_DIS	SDR104_OE_DIS	SDR104_TIMING_
W									CLR_DIS				TEST			DIS
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0

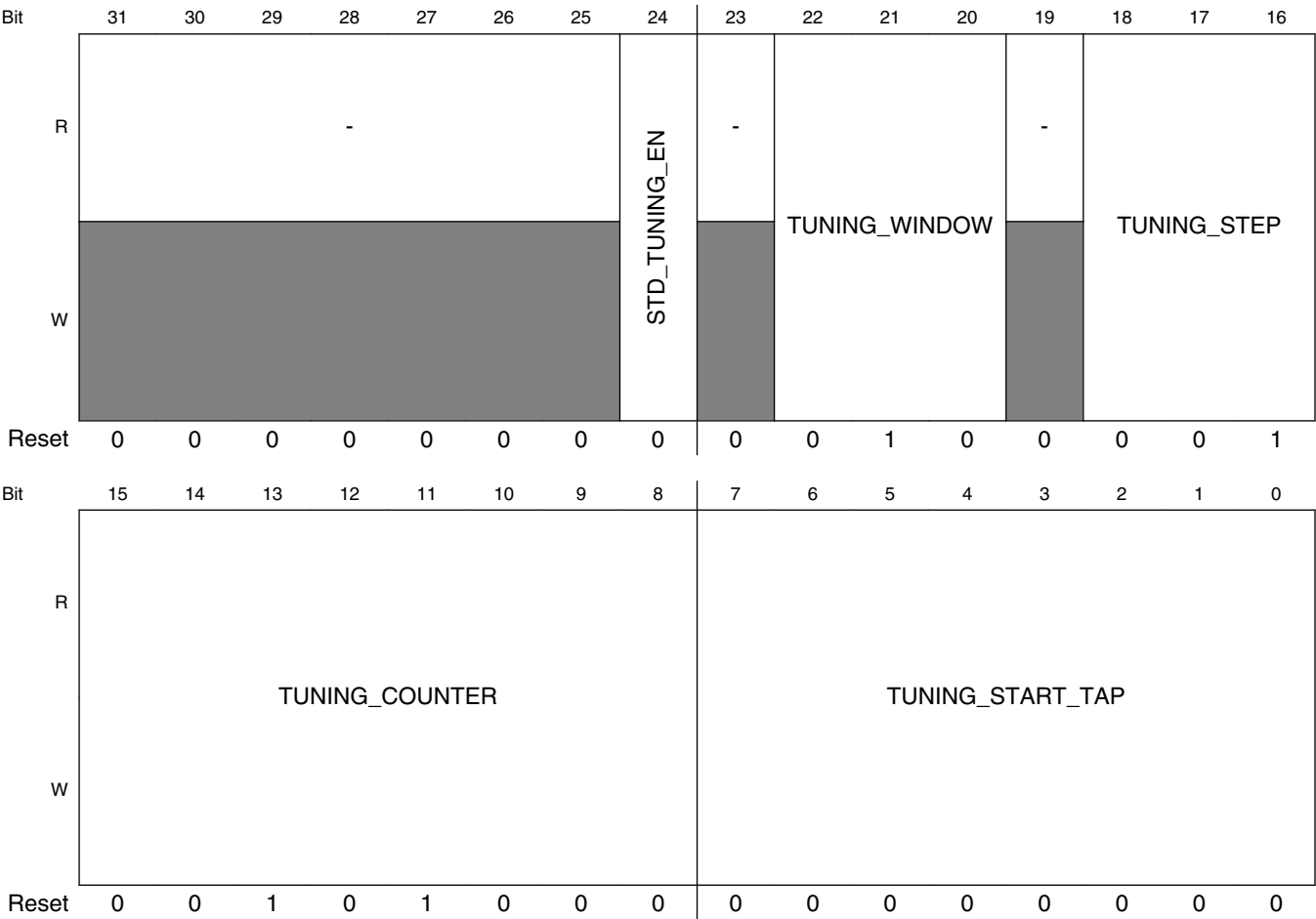
**uSDHCx\_VEND\_SPEC2 field descriptions**

Field	Description
31–8 Reserved	This read-only field is reserved and always has the value 0.
7 CARD_INT_ AUTO_CLR_DIS	Disable the feature to clear the Card interrupt status bit when Card Interrupt status enable bit is cleared. Only for debug.  0 Card interrupt status bit(CINT) can be cleared when Card Interrupt status enable bit is 0. 1 Card interrupt status bit(CINT) can only be cleared by writting a 1 to CINT bit.
6 TUNING_CMD_ EN	Enable the auto tuning circuit to check the CMD line.  0 Auto tuning circuit doesn't check the CMD line. 1 Auto tuning circuit checks the CMD line.
5 TUNING_1bit_EN	Enable the auto tuning circuit to check the DATA0 only. It's used with the TUNING_8bit_EN together.
4 TUNING_8bit_EN	Enable the auto tuning circuit to check the DATA[7:0]. It's used with the TUNING_1bit_EN together.  00 Tuning circuit only checks the DATA[3:0]. 01 Tuning circuit only checks the DATA0. 10 Tuning circuit checks the whole DATA[7:0]. 11 Invalid.
3 CARD_INT_D3_ TEST	Card interrupt detection test. This bit only uses for debugging.  0 Check the card interrupt only when DATA3 is high. 1 Check the card interrupt by ignoring the status of DATA3.
2 SDR104_NSD_ DIS	Interrupt window after abort command is sent. This bit only uses for debugging.  0 Enable the interrupt window 9 cycles later after the end of the I/O abort command(or CMD12) is sent. 1 Enable the interrupt window 5 cycles later after the end of the I/O abort command(or CMD12) is sent.
1 SDR104_OE_ DIS	CMD_OE/DATA_OE logic generation test. This bit only uses for debugging.  0 Drive the CMD_OE/DATA_OE for one more clock cycle after the end bit. 1 Stop to drive the CMD_OE/DATA_OE at once after driving the end bit.
0 SDR104_ TIMING_DIS	Timeout counter test. This bit only uses for debugging.  0 The timeout counter for Ncr changes to 80, Ncrc changes to 21. 1 The timeout counter for Ncr changes to 72, Ncrc changes to 15.

54.8.30 Tuning Control Register (uSDHCx\_TUNING\_CTRL)

The register contains configuration of tuning circuit.

Address: Base address + CCh offset



uSDHCx\_TUNING\_CTRL field descriptions

Field	Description
31–25 -	Reserved
24 STD_TUNING_EN	Standard tuning circuit and procedure enable: This bit is used to enable standard tuning circuit and procedure.
23 -	Reserved
22–20 TUNING_WINDOW	Select data window value for auto tuning
19 -	Reserved

Table continues on the next page...



**uSDHCx\_TUNING\_CTRL field descriptions (continued)**

<b>Field</b>	<b>Description</b>
18–16 TUNING_STEP	The increasing delay cell step in tuning procedure.
15–8 TUNING_COUNTER	The MAX repeat CMD19 times in tuning procedure.
7–0 TUNING_START_TAP	The start dealy cell point when send first CMD19 in tuning procedure.



## Chapter 55

# Watchdog Timer (WDOG)

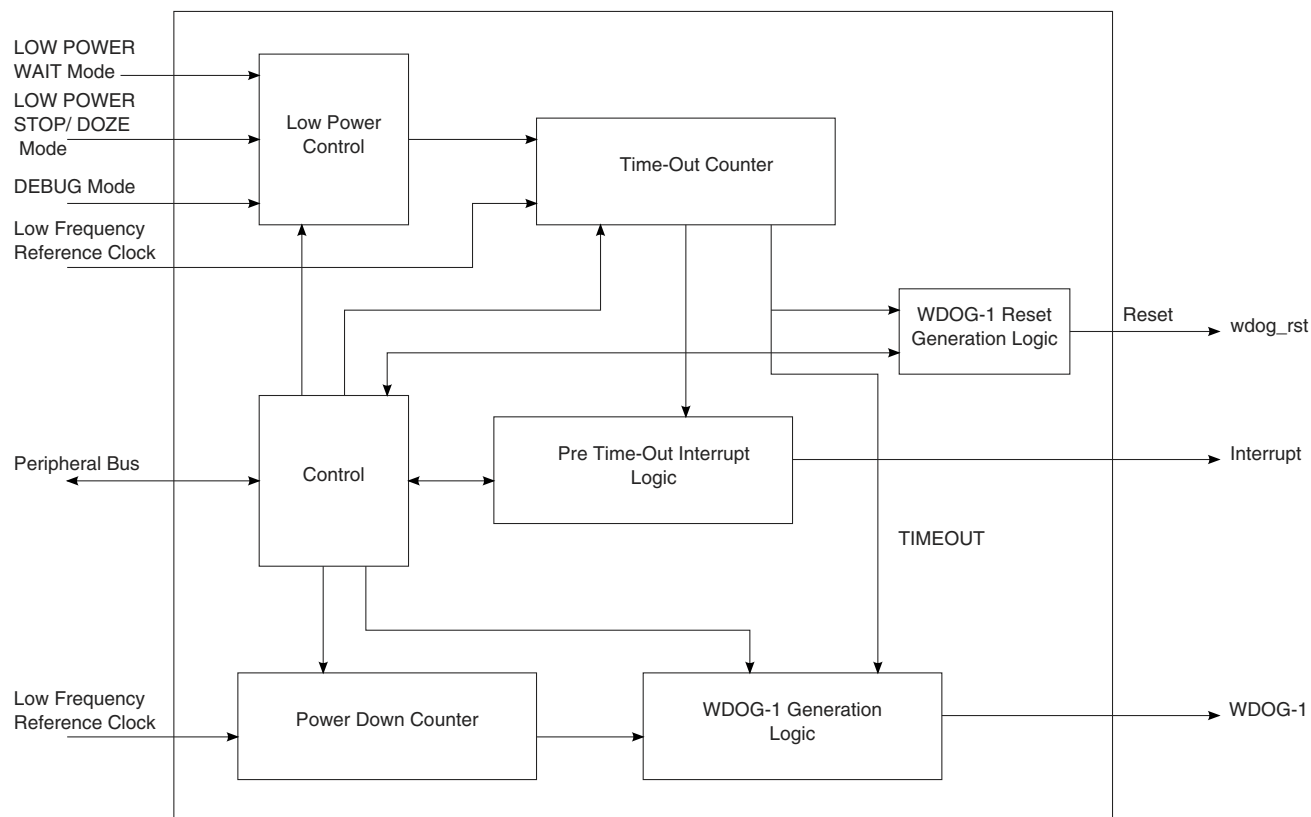
### 55.1 Overview

The Watchdog Timer (WDOG) protects against system failures by providing a method by which to escape from unexpected events or programming errors.

Once the WDOG is activated, it must be serviced by the software on a periodic basis. If servicing does not take place, the timer times out. Upon timeout, the WDOG asserts the internal system reset signal, WDOG\_RESET\_B\_DEB to the System Reset Controller (SRC).

There is also a provision for WDOG signal assertion by timeout counter expiration. There is an option of programmable interrupt generation before the counter actually times out. The time at which the interrupt needs to be generated prior to counter timeout is programmable. There is a power down counter which is enabled out of any reset (POR, Warm/Cold). This counter has a fixed timeout period of 16 seconds, upon which it asserts the WDOG signal.

Flow diagrams for the timeout counter, power down counter and interrupt operations are shown in [Figure 55-7](#), [Figure 55-8](#) and [Figure 55-9](#).



**Figure 55-1. WDOG Diagram**

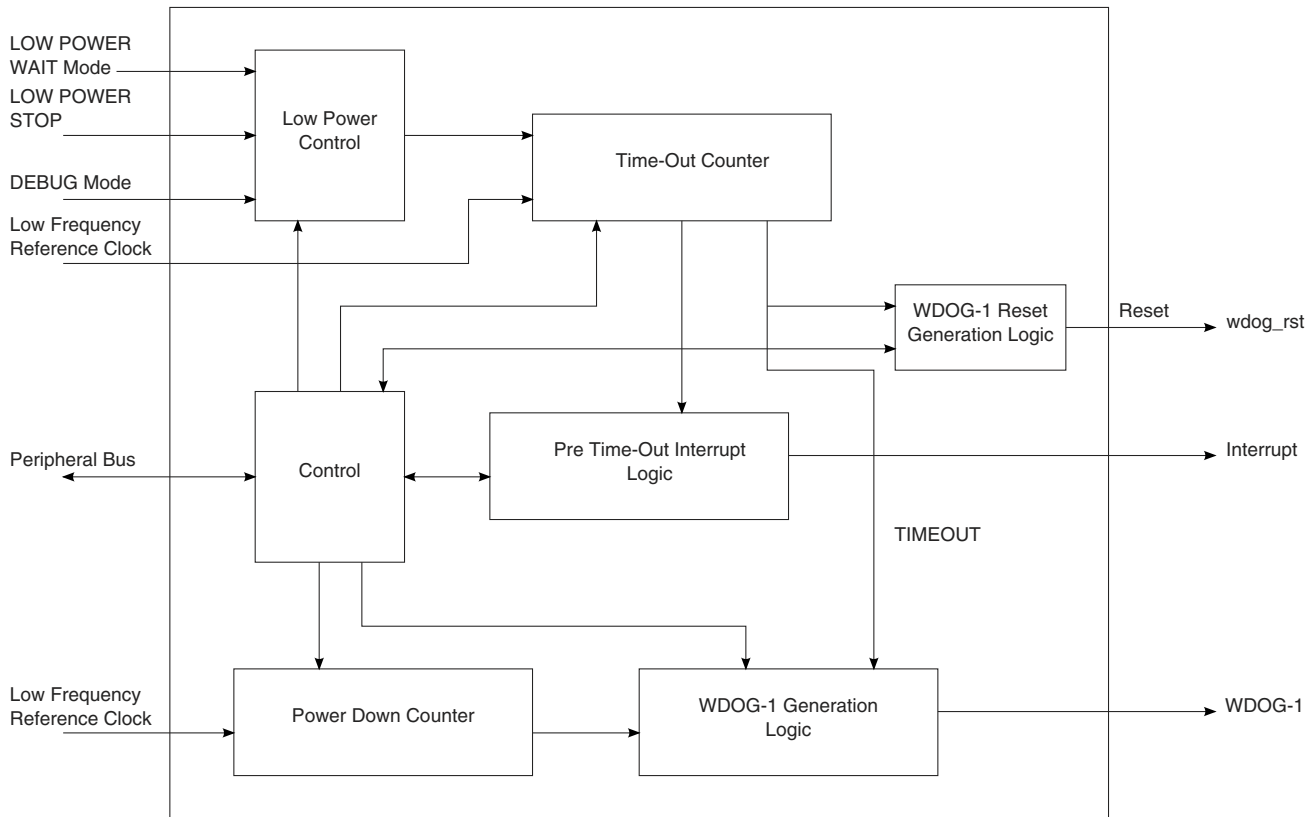


Figure 55-2.

### 55.1.1 Features

The WDOG features are listed below:

- Configurable timeout counter with timeout periods from 0.5 to 128 seconds which, after timeout expiration, result in the assertion of WDOG\_RESET\_B\_DEB reset signal .
- Time resolution of 0.5 seconds
- Configurable timeout counter that can be programmed to run or stop during low-power modes
- Configurable timeout counter that can be programmed to run or stop during DEBUG mode
- Programmable interrupt generation prior to timeout
- The duration between interrupt and timeout events can be programmed from 0 to 127.5 seconds in steps of 0.5 seconds.
- Power down counter with fixed timeout period of 16 seconds, which if not disabled after reset will assert WDOG\_B signal low

- Power down counter will be enabled out of any reset (POR, Warm / Cold reset) by default.

## 55.2 External signals

**Table 55-1. WDOG External Signals**

Signal	Description	Pad	Mode	Direction
WDOG1_B	This signal will power down the chip.	FEC_REF_CLK	ALT2	I/O
		WDOG_B	ALT0	
WDOG1_RESET_B_D EB	This signal is a reset source for the chip.	SD3_CLK	ALT4	O
		WDOG_B	ALT1	
WDOG2_B	This signal will power down the chip.	EPDC_SDCE1	ALT1	I/O
		SD2_RST	ALT2	
WDOG2_RESET_B_D EB	This signal is a reset source for the chip.	AUD_MCLK	ALT4	O
		LCD_DAT4	ALT3	

## 55.3 Clocks

This section describes clocks and special clocking requirements of the block.

The WDOG uses the low frequency reference clock for its counter and control operations. The peripheral bus clock is used for register read/write operations.

The following table describes the clock sources for WDOG. Please see [Clock Controller Module \(CCM\)](#) for clock setting, configuration and gating information.

**Table 55-2. WDOG Clocks**

Clock name	Clock Root	Description
ipg_clk	ipg_clk_root	IP Global functional clock. All functionality inside the WDOG module is synchronized to this clock.
ipg_clk_s	ipg_clk_root	IP slave bus clock. This clock is synchronized to ipg_clk and is only used for register read/write operations.
ipg_clk_32k	ckil_sync_clk_root	Low frequency (32.768 kHz) clock that continues to run in low-power mode. It is assumed that the Clock Controller will provide this clock signal synchronized to ipg_clk in the normal mode, and switch to a non-synchronized signal in low-power mode when the ipg_clk is off.

## 55.4 Functional description

This section provides a complete functional description of the block.

### 55.4.1 Timeout event

The WDOG provides timeout periods from 0.5 to 128 seconds with a time resolution of 0.5 seconds.

The user can determine the timeout period by writing to the WDOG timeout field (WT[7:0]) in the [Watchdog Control Register \(WDOG\\_WCR\)](#). The WDOG must be enabled by setting the WDE bit of [Watchdog Control Register \(WDOG\\_WCR\)](#) for the timeout counter to start running. After the WDOG is enabled, the counter is activated, loads the timeout value and begins to count down from this programmed value. The timer will time out when the counter reaches zero and the WDOG outputs a system reset signal, WDOG\_RESET\_B\_DEB and asserts WDOG\_B (WDT bit should be set in [Watchdog Control Register \(WDOG\\_WCR\)](#)).

However, the timeout condition can be prevented by reloading the counter with the new timeout value (WT[7:0] of WDOG\_WCR) if a service routine (see [Servicing WDOG to reload the counter](#)) is performed before the counter reaches zero. If any system errors occur which prevent the software from servicing the [Watchdog Service Register \(WDOG\\_WSR\)](#), the timeout condition occurs. By performing the service routine, the WDOG reloads its counter to the timeout value indicated by bits WT[7:0] of the [Watchdog Control Register \(WDOG\\_WCR\)](#) and it restarts the countdown.

A system reset will reset the counter and place it in the idle state at any time during the countdown. The counter flow diagram is shown in [Figure 55-7](#).

#### NOTE

The timeout value is reloaded to the counter either at the time WDOG is enabled or after the service routine has been performed.

#### 55.4.1.1 Servicing WDOG to reload the counter

To reload a timeout value to the counter the proper service sequence begins by writing 0x-5555 followed by 0x-AAAA to the [Watchdog Service Register \(WDOG\\_WSR\)](#). Any number of instructions can be executed between the two writes. If the WDOG\_WSR is not loaded with 0x-5555 prior to writing 0x-AAAA to the WDOG\_WSR, the counter is not reloaded. If any value other than 0x-AAAA is written to the WDOG\_WSR after

0x-5555, the counter is not reloaded. This service sequence will reload the counter with the timeout value WT[7:0] of [Watchdog Control Register \(WDOG\\_WCR\)](#). The timeout value can be changed at any point; it is reloaded when WDOG is serviced by the core.

## **55.4.2 Interrupt event**

Prior to timeout, the WDOG can generate an interrupt which can be considered a warning that timeout will occur shortly.

The duration between interrupt event and timeout event can be controlled by writing to the WICT field of [Watchdog Interrupt Control Register \(WDOG\\_WICR\)](#). It can vary between 0 and 127.5 seconds. If the WDOG is serviced ([Servicing WDOG to reload the counter](#)) before the interrupt generation, the counter will be reloaded with the timeout value WT[7:0] of [Watchdog Control Register \(WDOG\\_WCR\)](#) and the interrupt will not be triggered.

## **55.4.3 Power-down counter event**

The power-down counter inside WDOG will be enabled out of reset. This counter has a fixed timeout value of 16 seconds, after which it will drive the WDOG\_B signal low.

To prevent this, the software must disable this counter by clearing the PDE bit of [Watchdog Miscellaneous Control Register \(WDOG\\_WMCR\)](#) within 16 seconds of reset deassertion. Once disabled, this counter can't be enabled again until the next system reset occurs. This feature is intended to prevent the hanging up of cores after reset, as WDOG is not enabled out of reset.

## **55.4.4 Low power modes**

### **55.4.4.1 STOP and DOZE mode**

If the WDOG timer disable bit for low power STOP and DOZE mode (WDZST) bit in the [Watchdog Control Register \(WDOG\\_WCR\)](#), is cleared, the WDOG timer continues to operate using the low frequency reference clock. If the low power enable (WDZST) bit is set, the WDOG timer operation will be suspended in low power STOP or DOZE mode. Upon exiting low power STOP or DOZE mode, the WDOG operation returns to what it was prior to entering the STOP or DOZE mode.



#### 55.4.4.2 WAIT mode

If the WDOG timer disable bit for low power WAIT mode (WDW) bit in the [Watchdog Control Register \(WDOG\\_WCR\)](#), is cleared, the WDOG timer continues to operate using the low frequency reference clock. If the low power WAIT enable (WDW) bit is set, the WDOG timer operation will be suspended. Upon exiting low power WAIT mode, the WDOG operation returns to what it was prior to entering the WAIT mode.

#### NOTE

The WDOG timer won't be able to detect events that happen for periods shorter than one low frequency reference clock cycle. For example, in repeated WAIT mode entry or exit, if the RUN mode time is less than one low frequency reference clock cycle and if the WDW bit is set, the WDOG timer may never time out, even though the system is in RUN mode for a finite duration; WDOG may not see a low frequency reference clock edge during its wake time.

#### 55.4.5 Debug mode

The WDOG timer can be configured for continual operation, or for suspension during debug mode. If the WDOG debug enable (WDBG) bit is set in the [Watchdog Control Register \(WDOG\\_WCR\)](#), the WDOG timer operation is suspended in debug mode. If the WDBG bit is set and the debug mode is entered, WDOG timer operation is suspended after two low frequency reference clocks. Similarly, WDOG timer operation continues after two low frequency reference clocks of debug mode exit. Register read and write accesses in debug mode continue to function normally. Also, while in debug mode, the WDE bit of [Watchdog Control Register \(WDOG\\_WCR\)](#) can be enabled/disabled directly. If the WDOG debug enable (WDBG) bit is cleared then WDOG timer operation is not suspended. The power-down counter is not affected by debug mode entry/exit.

#### NOTE

If the WDE bit of [Watchdog Control Register \(WDOG\\_WCR\)](#) is set/cleared while in debug mode, it remains set/cleared even after exiting debug mode.

#### 55.4.6 Operations

### 55.4.6.1 Watchdog reset generation

The WDOG generated reset signal WDOG\_RESET\_B\_DEB is asserted by the following operations:

- A software write to the Software Reset Signal (SRS) bit of the [Watchdog Control Register \(WDOG\\_WCR\)](#).
- WDOG timeout. See [Timeout event](#).

The  $\overline{\text{wdog\_rst}}$  will be asserted for one clock cycle of low frequency reference clock for both a timeout condition and a software write occurrence. It remains asserted for 1 clock cycle of low frequency reference clock even if a system reset is asserted in between.

[Figure 55-4](#) shows the timing diagram of this signal due to a timeout condition.

### 55.4.6.2 WDOG\_B generation

The WDOG asserts WDOG\_B in the following scenarios:

- Software write to WDA bit of [Watchdog Control Register \(WDOG\\_WCR\)](#). WDOG\_B signal remains asserted as long as the WDA bit is "0".
- WDOG timeout condition, WDT bit of [Watchdog Control Register \(WDOG\\_WCR\)](#) must be set for this scenario. A description of the timeout condition can be found in the [Timeout event](#). WDOG\_B signal remains asserted until a power-on reset (POR) occurs. It gets cleared after the POR occurs (not due to any other system reset). [Figure 55-5](#) shows the timing diagram of WDOG\_B due to timeout condition.
- WDOG power-down counter timeout, PDE bit of [Watchdog Miscellaneous Control Register \(WDOG\\_WMCR\)](#) should not be cleared for this scenario. A description of this counter can be found in the [Power-down counter event](#). WDOG\_B signal remains asserted for one clock cycle of low frequency reference clock.

[Figure 55-3](#) shows the scenarios under which WDOG\_B gets asserted.

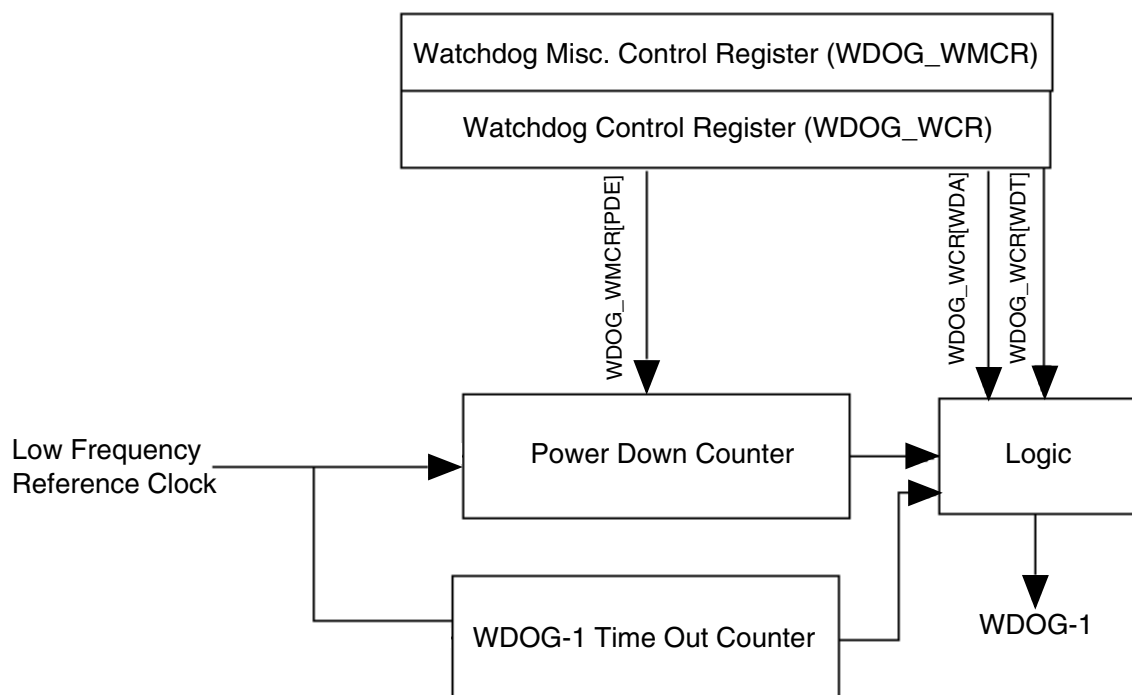


Figure 55-3. WDOG\_B generation

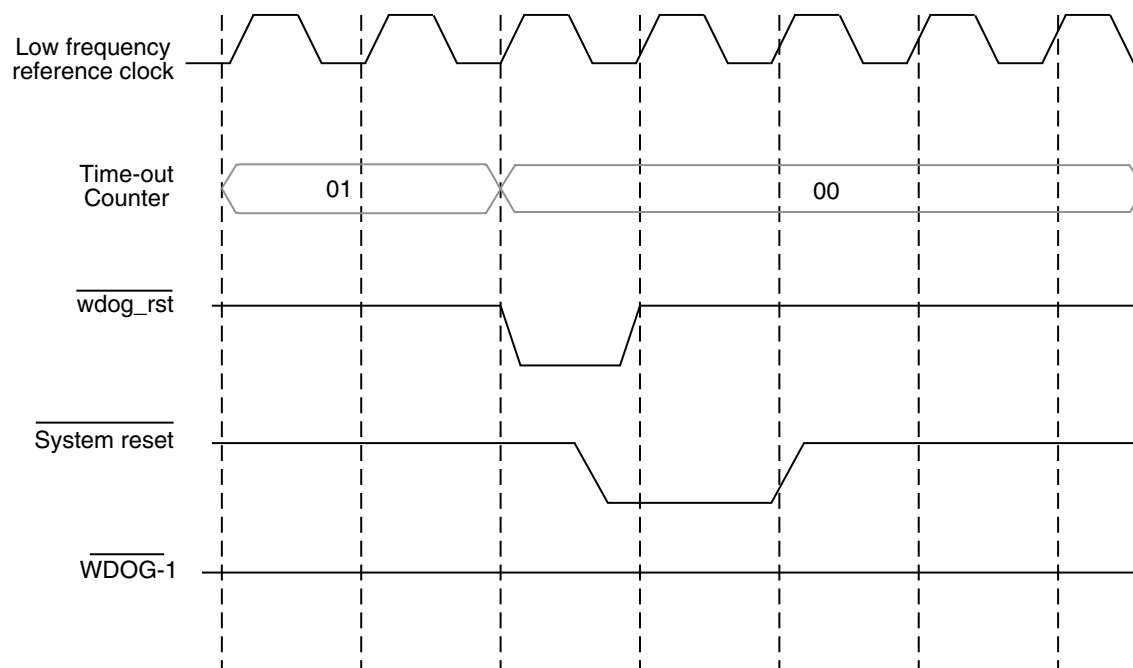
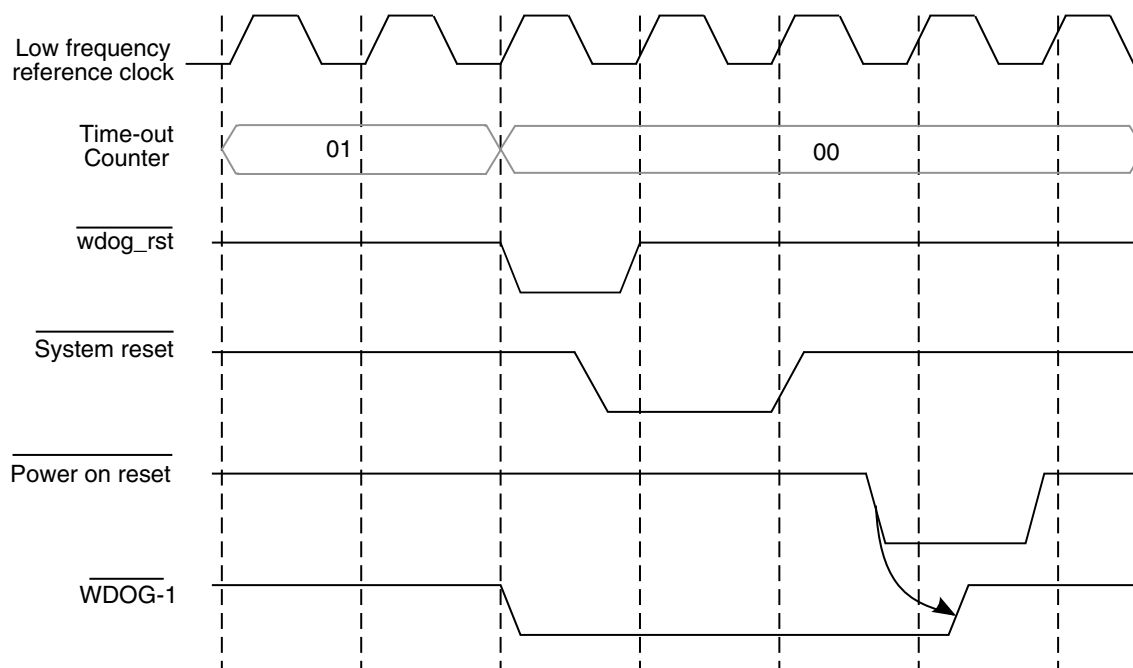


Figure 55-4. WDOG timeout condition/WDT bit is not set



**Figure 55-5. WDOG timeout condition/WDT bit is set**

### 55.4.7 Reset

The block is reset by a system reset and the WDOG counter will be disabled. The power-down counter is enabled and starts counting.

### 55.4.8 Interrupt

The WDOG has the feature of Interrupt generation before timeout.

The interrupt will be generated only if the WIE bit in [Watchdog Interrupt Control Register \(WDOG\\_WICR\)](#) is set. The exact time at which the interrupt should occur (prior to timeout) depends on the value of WICT field of [Watchdog Interrupt Control Register \(WDOG\\_WICR\)](#). For example, if the WICT field has a value 0x04, then the interrupt will be generated two seconds prior to timeout. Once the interrupt is triggered the WTIS bit in [Watchdog Interrupt Control Register \(WDOG\\_WICR\)](#) will be set. The software needs to clear this bit to deassert the interrupt. If the WDOG is serviced before the interrupt generation then the counter will be reloaded with the timeout value WT[7:0] of [Watchdog Control Register \(WDOG\\_WCR\)](#) and interrupt would not be triggered.

## 55.4.9 Flow Diagrams

A flow diagram of WDOG operation is shown in [Figure 55-7](#), [Figure 55-8](#) and [Figure 55-9](#).

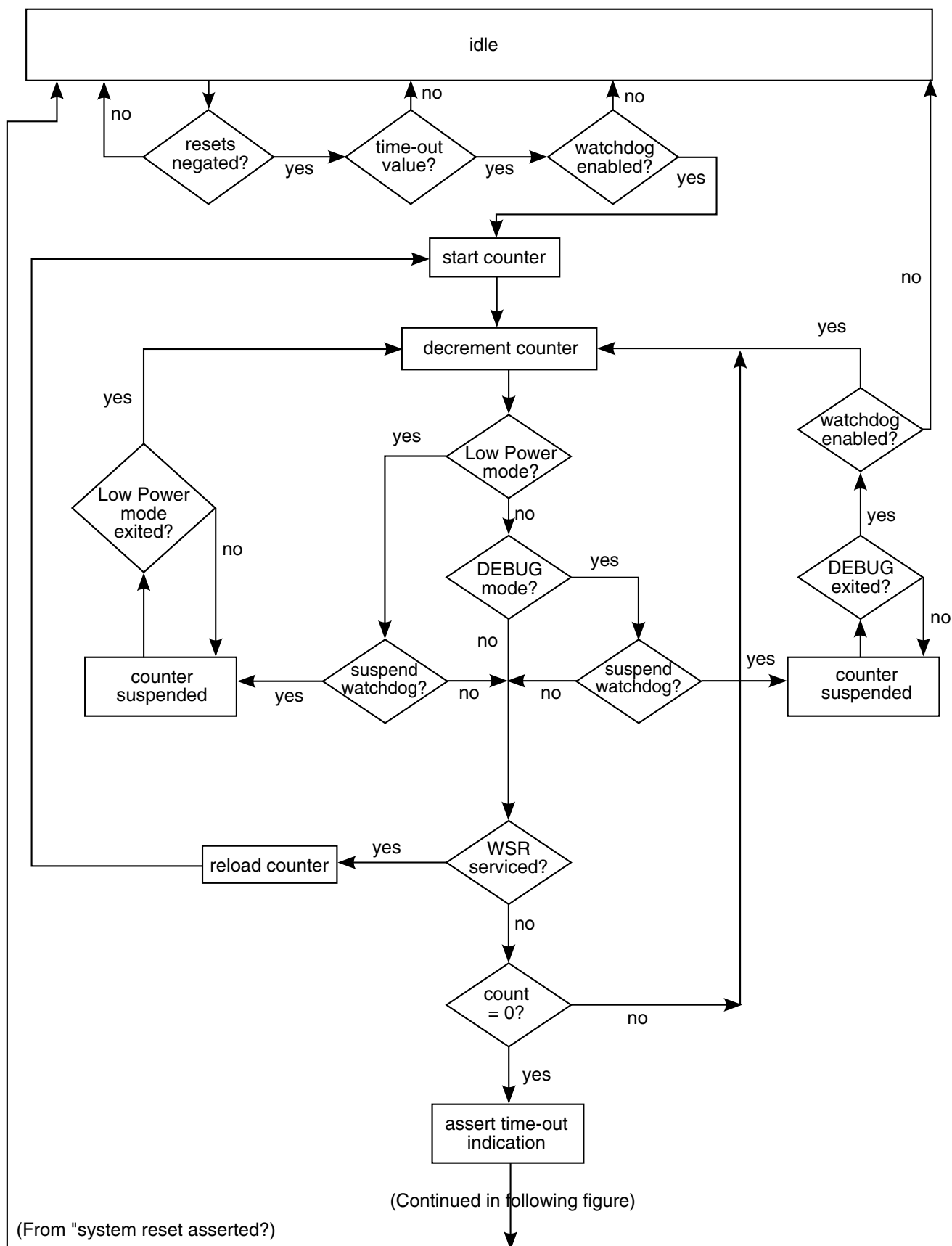
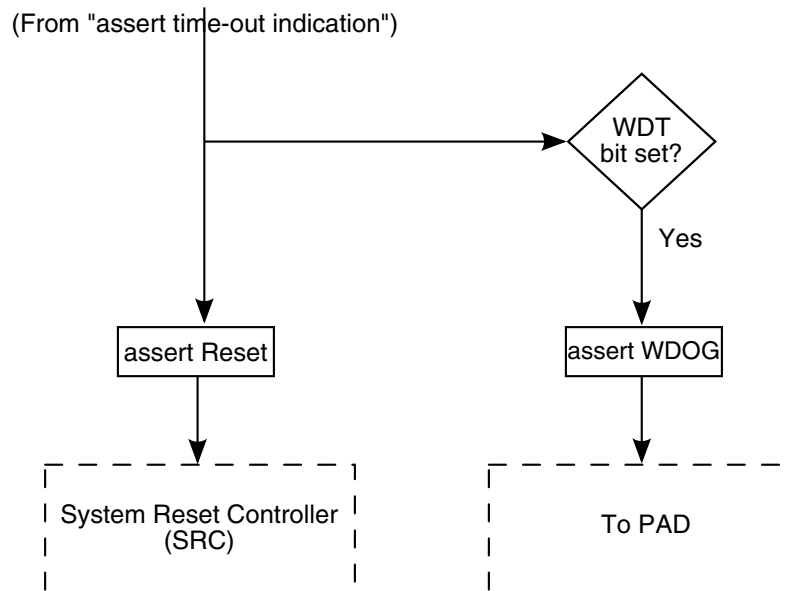
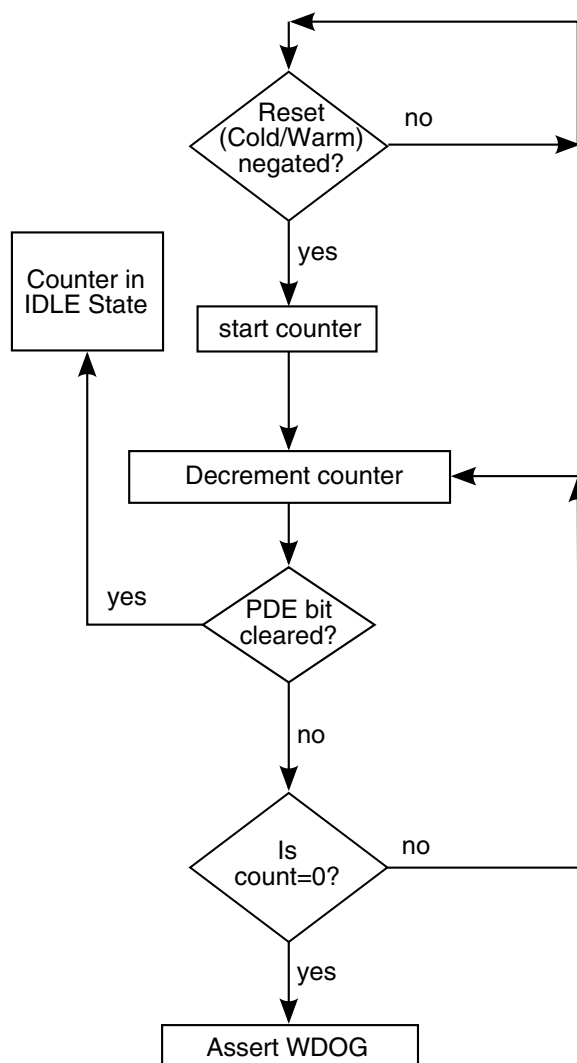


Figure 55-6. Time-Out Counter Flow Diagram



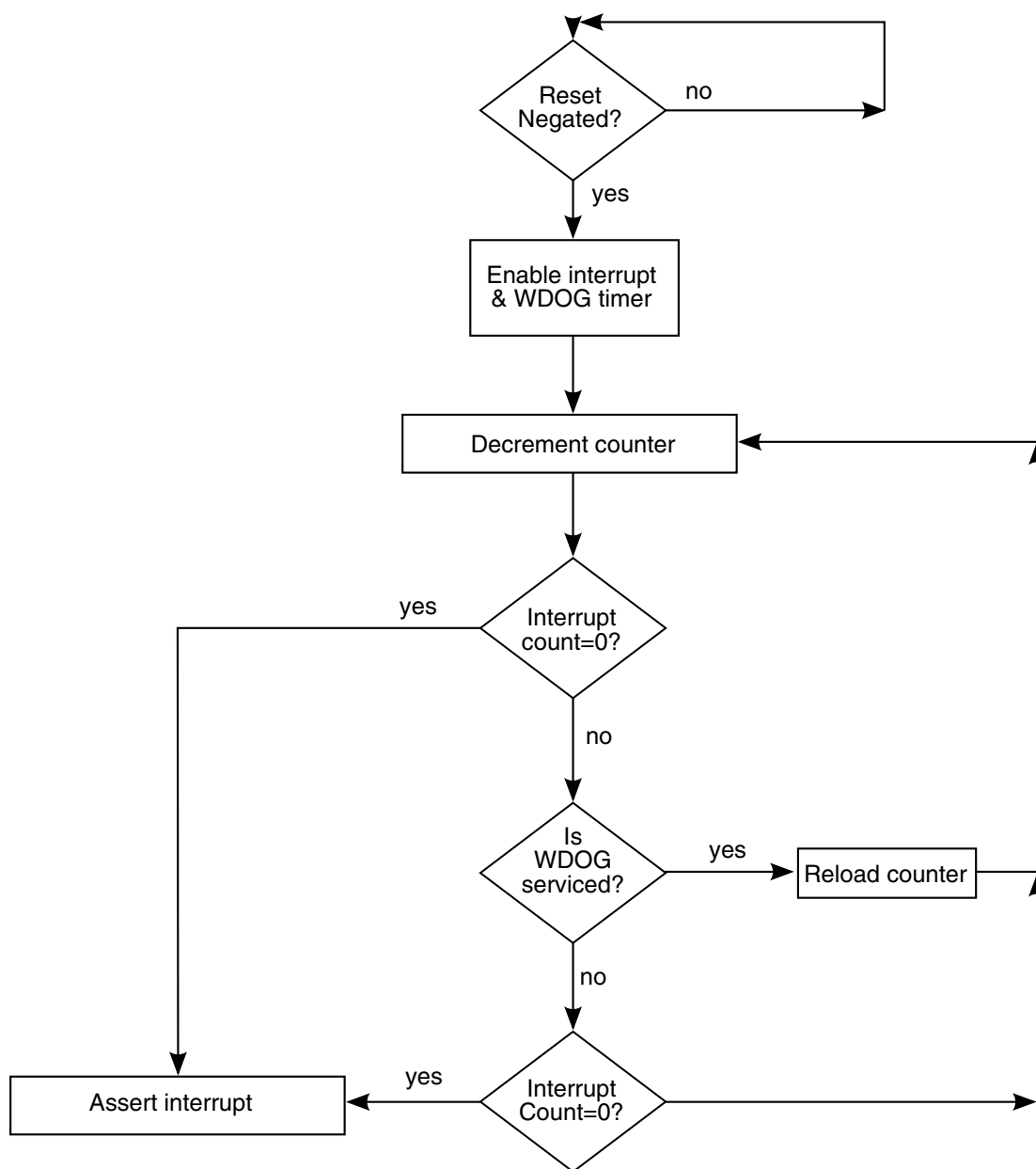
Note: A system reset will force the state machine to "idle" at any time during countdown.

**Figure 55-7. Time-Out Counter Flow Diagram**



**Figure 55-8. Power-Down Counter Flow Diagram**





**Figure 55-9. Interrupt Generation Flow Diagram**

## 55.5 Initialization

The following sequence should be performed for WDOG initialization.

- PDE bit of [Watchdog Miscellaneous Control Register \(WDOG\\_WMCR\)](#) should be cleared to disable the power down counter.

- WDT field of [Watchdog Control Register \(WDOG\\_WCR\)](#) should be programmed for sufficient timeout value.
- WDOG should be enabled by setting WDE bit of [Watchdog Control Register \(WDOG\\_WCR\)](#) so that the timeout counter loads the WDT field value of [Watchdog Control Register \(WDOG\\_WCR\)](#) and starts counting.

## 55.6 WDOG Memory Map/Register Definition

The WDOG has user-accessible, 16-bit registers used to configure, operate, and monitor the state of the Watchdog Timer. Byte operations can be performed on these registers. If a 32-bit access is performed, the WDOG will not generate a peripheral bus error but will behave normally, like a 16-Bit access, making read/write possible. A 32-Bit access should be avoided, as the system may go to an unknown state.

**WDOG memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/page
20B_C000	Watchdog Control Register (WDOG1_WCR)	16	R/W	0030h	<a href="#">55.6.1/3462</a>
20B_C002	Watchdog Service Register (WDOG1_WSR)	16	R/W	0000h	<a href="#">55.6.2/3464</a>
20B_C004	Watchdog Reset Status Register (WDOG1_WRSR)	16	R	0000h	<a href="#">55.6.3/3465</a>
20B_C006	Watchdog Interrupt Control Register (WDOG1_WICR)	16	R/W	0004h	<a href="#">55.6.4/3466</a>
20B_C008	Watchdog Miscellaneous Control Register (WDOG1_WMCR)	16	R/W	0001h	<a href="#">55.6.5/3467</a>
20C_0000	Watchdog Control Register (WDOG2_WCR)	16	R/W	0030h	<a href="#">55.6.1/3462</a>
20C_0002	Watchdog Service Register (WDOG2_WSR)	16	R/W	0000h	<a href="#">55.6.2/3464</a>
20C_0004	Watchdog Reset Status Register (WDOG2_WRSR)	16	R	0000h	<a href="#">55.6.3/3465</a>
20C_0006	Watchdog Interrupt Control Register (WDOG2_WICR)	16	R/W	0004h	<a href="#">55.6.4/3466</a>
20C_0008	Watchdog Miscellaneous Control Register (WDOG2_WMCR)	16	R/W	0001h	<a href="#">55.6.5/3467</a>

### 55.6.2 Watchdog Control Register (WDOGx\_WCR)

The Watchdog Control Register (WDOG\_WCR) controls the WDOG operation.

- WDZST, WDBG and WDW are write-once only bits. Once the software does a write access to these bits, they will be locked and cannot be reprogrammed until the next system reset assertion.

- WDE is a write one once only bit. Once software performs a write "1" operation to this bit it cannot be reset/cleared until the next system reset.
- WDT is also a write one once only bit. Once software performs a write "1" operation to this bit it cannot be reset/cleared until the next POR. This bit does not get reset/cleared due to any system reset.

Address: Base address + 0h offset

Bit	15	14	13	12	11	10	9	8
Read	WT							
Write								
Reset	0	0	0	0	0	0	0	0

Bit	7	6	5	4	3	2	1	0
Read	WDW	SRE	WDA	SRS	WDT	WDE	WDBG	WDZST
Write								
Reset	0	0	1	1	0	0	0	0

### WDOGx\_WCR field descriptions

Field	Description
15–8 WT	<p>Watchdog Time-out Field. This 8-bit field contains the time-out value that is loaded into the Watchdog counter after the service routine has been performed or after the Watchdog is enabled. After reset, WT[7:0] must have a value written to it before enabling the Watchdog otherwise count value of zero which is 0.5 seconds is loaded into the counter.</p> <p><b>NOTE:</b> The time-out value can be written at any point of time but it is loaded to the counter at the time when WDOG is enabled or after the service routine has been performed. For more information see <a href="#">Timeout event</a> .</p> <p>0x00 - 0.5 Seconds (Default).  0x01 - 1.0 Seconds.  0x02 - 1.5 Seconds.  0x03 - 2.0 Seconds.  0xff - 128 Seconds.</p>
7 WDW	<p>Watchdog Disable for Wait. This bit determines the operation of WDOG during Low Power WAIT mode. This is a write once only bit.</p> <p>0 Continue WDOG timer operation (Default).  1 Suspend WDOG timer operation.</p>
6 SRE	<p>software reset extension, an option way to generate software reset</p> <p>adopt a new way to generate a more robust software reset. This bit can be set/clear with IP bus and will be reset with power-on reset .</p> <p>0 using original way to generate software reset (default)  1 using new way to generate software reset.</p>
5 WDA	<p>WDOG_B assertion. Controls the software assertion of the WDOG_B signal.</p> <p>0 Assert WDOG_B output.  1 No effect on system (Default).</p>
4 SRS	<p>Software Reset Signal. Controls the software assertion of the WDOG-generated reset signal WDOG_RESET_B_DEB . This bit automatically resets to "1" after it has been asserted to "0".</p> <p><b>NOTE:</b> This bit does not generate the software reset to the block.</p>

Table continues on the next page...

## WDOGx\_WCR field descriptions (continued)

Field	Description
	0 Assert system reset signal. 1 No effect on the system (Default).
3 WDT	WDOG_B Time-out assertion. Determines if the WDOG_B gets asserted upon a Watchdog Time-out Event. This is a write-one once only bit.  <b>NOTE:</b> There is no effect on WDOG_RESET_B_DEB (WDOG Reset) upon writing on this bit. WDOG_B gets asserted along with WDOG_RESET_B_DEB if this bit is set.  0 No effect on WDOG_B (Default). 1 Assert WDOG_B upon a Watchdog Time-out event.
2 WDE	Watchdog Enable. Enables or disables the WDOG block. This is a write one once only bit. It is not possible to clear this bit by a software write, once the bit is set.  <b>NOTE:</b> This bit can be set/reset in debug mode (exception).  0 Disable the Watchdog (Default). 1 Enable the Watchdog.
1 WDBG	Watchdog DEBUG Enable. Determines the operation of the WDOG during DEBUG mode. This bit is write once only.  0 Continue WDOG timer operation (Default). 1 Suspend the watchdog timer.
0 WDZST	Watchdog Low Power. Determines the operation of the WDOG during low-power modes. This bit is write once-only.  <b>NOTE:</b> The WDOG can continue/suspend the timer operation in the low-power modes (STOP and DOZE mode).  0 Continue timer operation (Default). 1 Suspend the watchdog timer.

## 55.6.3 Watchdog Service Register (WDOGx\_WSR)

When enabled, the WDOG requires that a service sequence be written to the Watchdog Service Register (WSR) to prevent the timeout condition.

**NOTE**

Executing the service sequence will reload the WDOG timeout counter.

Address: Base address + 2h offset

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	WSR															
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

**WDOGx\_WSR field descriptions**

Field	Description
15–0 WSR	<p>Watchdog Service Register. This 16-bit field contains the Watchdog service sequence. Both writes must occur in the order listed prior to the time-out, but any number of instructions can be executed between the two writes. The service sequence must be performed as follows:</p> <p>0x5555 Write to the Watchdog Service Register (WDOG_WSR).</p> <p>0xAAAA Write to the Watchdog Service Register (WDOG_WSR).</p>

**55.6.4 Watchdog Reset Status Register (WDOGx\_WRSR)**

The WRSR is a read-only register that records the source of the output reset assertion. It is not cleared by a hard reset. Therefore, only one bit in the WRSR will always be asserted high. The register will always indicate the source of the last reset generated due to WDOG. Read access to this register is with one wait state. Any write performed on this register will generate a Peripheral Bus Error .

A reset can be generated by the following sources, as listed in priority from highest to lowest:

- Watchdog Time-out
- Software Reset

Address: Base address + 4h offset

Bit	15	14	13	12	11	10	9	8
Read	0							
Write								
Reset	0	0	0	0	0	0	0	0
Bit	7	6	5	4	3	2	1	0
Read	0			POR	0		TOUT	SFTW
Write								
Reset	0	0	0	0	0	0	0	0

**WDOGx\_WRSR field descriptions**

Field	Description
15–5 Reserved	This read-only field is reserved and always has the value 0.
4 POR	<p>Power On Reset. Indicates whether the reset is the result of a power on reset.</p> <p>0 Reset is not the result of a power on reset.</p> <p>1 Reset is the result of a power on reset.</p>
3–2 Reserved	This read-only field is reserved and always has the value 0.

*Table continues on the next page...*

## WDOGx\_WRSR field descriptions (continued)

Field	Description
1 TOUT	Timeout. Indicates whether the reset is the result of a WDOG timeout. 0 Reset is not the result of a WDOG timeout. 1 Reset is the result of a WDOG timeout.
0 SFTW	Software Reset. Indicates whether the reset is the result of a WDOG software reset by asserting SRS bit 0 Reset is not the result of a software reset. 1 Reset is the result of a software reset.

## 55.6.5 Watchdog Interrupt Control Register (WDOGx\_WICR)

The WDOG\_WICR controls the WDOG interrupt generation.

Address: Base address + 6h offset

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	WIE	WTIS	0						WICT							
Write		w1c														
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0

## WDOGx\_WICR field descriptions

Field	Description
15 WIE	Watchdog Timer Interrupt enable bit. Reset value is 0.  <b>NOTE:</b> This bit is a write once only bit. Once the software does a write access to this bit, it will get locked and cannot be reprogrammed until the next system reset assertion  0 Disable Interrupt (Default). 1 Enable Interrupt.
14 WTIS	Watchdog Timer Interrupt Status bit will reflect the timer interrupt status, whether interrupt has occurred or not. Once the interrupt has been triggered software must clear this bit by writing 1 to it.  0 No interrupt has occurred (Default). 1 Interrupt has occurred
13–8 Reserved	This read-only field is reserved and always has the value 0.
7–0 WICT	Watchdog Interrupt Count Time-out (WICT) field determines, how long before the counter time-out must the interrupt occur. The reset value is 0x04 implies interrupt will occur 2 seconds before time-out. The maximum value that can be programmed to WICT field is 127.5 seconds with a resolution of 0.5 seconds.  <b>NOTE:</b> This field is write once only. Once the software does a write access to this field, it will get locked and cannot be reprogrammed until the next system reset assertion.  0x00 WICT[7:0] = Time duration between interrupt and time-out is 0 seconds. 0x01 WICT[7:0] = Time duration between interrupt and time-out is 0.5 seconds. 0x04 WICT[7:0] = Time duration between interrupt and time-out is 2 seconds (Default). 0xff WICT[7:0] = Time duration between interrupt and time-out is 127.5 seconds.

### 55.6.6 Watchdog Miscellaneous Control Register (WDOGx\_WMCR)

WDOG\_WMCR Controls the Power Down counter operation.

Address: Base address + 8h offset

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Read	0															PDE
Write																
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

WDOGx\_WMCR field descriptions

Field	Description
15–1 Reserved	This read-only field is reserved and always has the value 0.
0 PDE	<p>Power Down Enable bit. Reset value of this bit is 1, which means the power down counter inside the WDOG is enabled after reset. The software must write 0 to this bit to disable the counter within 16 seconds of reset de-assertion. Once disabled this counter cannot be enabled again. See <a href="#">Power-down counter event</a> for operation of this counter.</p> <p><b>NOTE:</b> This bit is write-one once only bit. Once software sets this bit it cannot be reset until the next system reset.</p> <p>0 Power Down Counter of WDOG is disabled. 1 Power Down Counter of WDOG is enabled (Default).</p>





## Chapter 56

# Crystal Oscillator (XTALOSC)

### 56.1 Overview

This block comprises both the 24 MHz and 32 kHz implementation of a biased amplifier that when combined with a suitable external quartz crystal and external load capacitors, implements an oscillator.

The block includes means to:

- Accept an external clock source
- Detect if the crystal frequency is close to 24 MHz or 32 kHz
- Reduce the operating current via software after the oscillator has started (24 MHz specific feature)
- Supply another ~32 kHz clock source based off an independent internal oscillator if there is no oscillation sensed on the RTC\_XTAL bumps(contacts) (32 kHz specific feature)
- Automatically switch to the external oscillation source when sensed on the RTC\_XTAL bumps(contacts) (32 kHz specific feature)

### 56.2 External Signals

The table found here describes the external signals of XTALOSC:

**Table 56-1. XTALOSC External Signals**

Signal	Description	Pad	Mode	Direction
CLK1_N	Negative differential clock signal	CLK1_N	No Muxing	O
CLK1_P	Positive differential clock signal	CLK1_P	No Muxing	O
REF_CLK_24M	24 MHz reference clock	REF_CLK_24M	ALT0	O
		HSIC_DAT	ALT3	

*Table continues on the next page...*

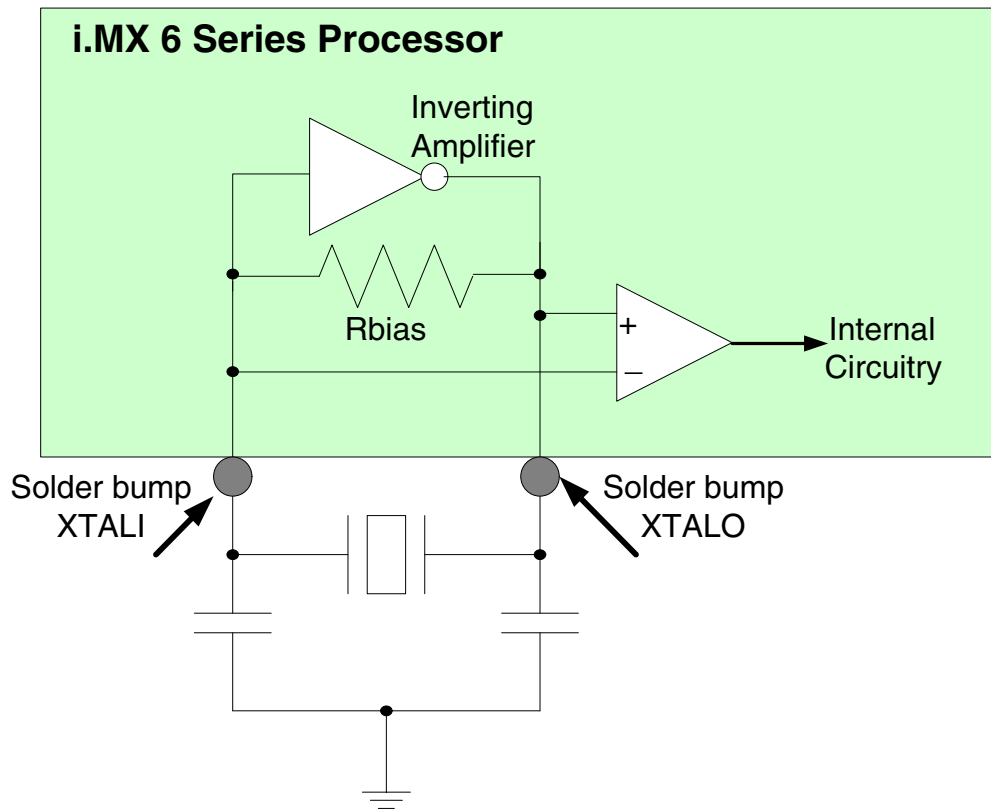
**Table 56-1. XTALOSC External Signals  
(continued)**

Signal	Description	Pad	Mode	Direction
REF_CLK_32K	32 KHz reference clock	REF_CLK_32K	ALT0	O
		HSIC_STROBE	ALT3	
RTC_XTALI	Real-time clock crystal oscillator input	RTC_XTALI	No Muxing	I
RTC_XTALO	Real-time clock crystal oscillator output	RTC_XTALO	No Muxing	O
XTALI	Crystal oscillator input signal	XTALI	No Muxing	I
XTALO	Crystal oscillator output signal	XTALO	No Muxing	O

## 56.3 Crystal Oscillator 24 MHz

### 56.3.1 Oscillator Configuration (24 MHz)

The basic block diagram of the 24 MHz module configured as a crystal oscillator is shown below.



**Figure 56-1. Oscillator Configuration (24 MHz)**

This integrated biased amplifier can be used to create different frequency oscillators with different external component selection. However, care should be taken as many of the serial IO modules depend on the fixed frequency of 24 MHz. Please consult the sections of the document pertaining to the USB, and ENET interfaces, for example. Once a healthy oscillation is established, then the bias current of the oscillator can generally be reduced to save power. This is accomplished through the XTALOSC24M\_MISC0[OSC\_I] bits, defined in the MISC0 register later in this chapter. Restore the XTALOSC24M\_MISC0[OSC\_I] bits before going into a power mode where the XTALOSC24 is powered down or oscillator startup may become an issue. The power down of the XTALOSC24 module is controlled by the CCM. See this section of the manual for more details.

### 56.3.2 Bypass Configuration (24 MHz)

If it is desired to drive the chip with an external clock source, then the 24 MHz oscillator could be driven in one of three configurations using a nominal 1.1V source.

1. A single ended external clock source can be used to overdrive the output of the amplifier (XTALO). Since the oscillation sensing amplifier is differential, the

XTALI pin should be externally floating and capacitively loaded. The combination of the internal biasing resistor and the external capacitor will filter the signal applied to the XTALO pin and develop a rough reference for the sensing amplifier to compare to.

2. A single ended external clock source can be used to drive XTALI. In this configuration, XTALO should be left externally floating.
3. A differential external clock source can be used to drive both XTALI and XTALO.

Generally, configuration 2 is anticipated to be the most used configuration, but all three configurations may be utilized.

### **56.3.3 RC Oscillator (24 MHz)**

A lower-power RC oscillator module is available on-chip as a possible alternative to the 24 MHz crystal oscillator after a successful power-up sequence. While the power consumption of this RC oscillator is much lower than the 24 MHz crystal oscillator, one limitation of this RC oscillator module is that its clock frequency is not as accurate. Therefore, care should be observed when using this oscillator as the reference for the on-chip PLLs as their output clock frequency will be lower/higher than when using the 24 MHz crystal oscillator clock. For more details on the possible usage of this module please contact a Freescale FAE for pertinent application-notes.

### **56.3.4 Crystal Frequency Detection(24 MHz)**

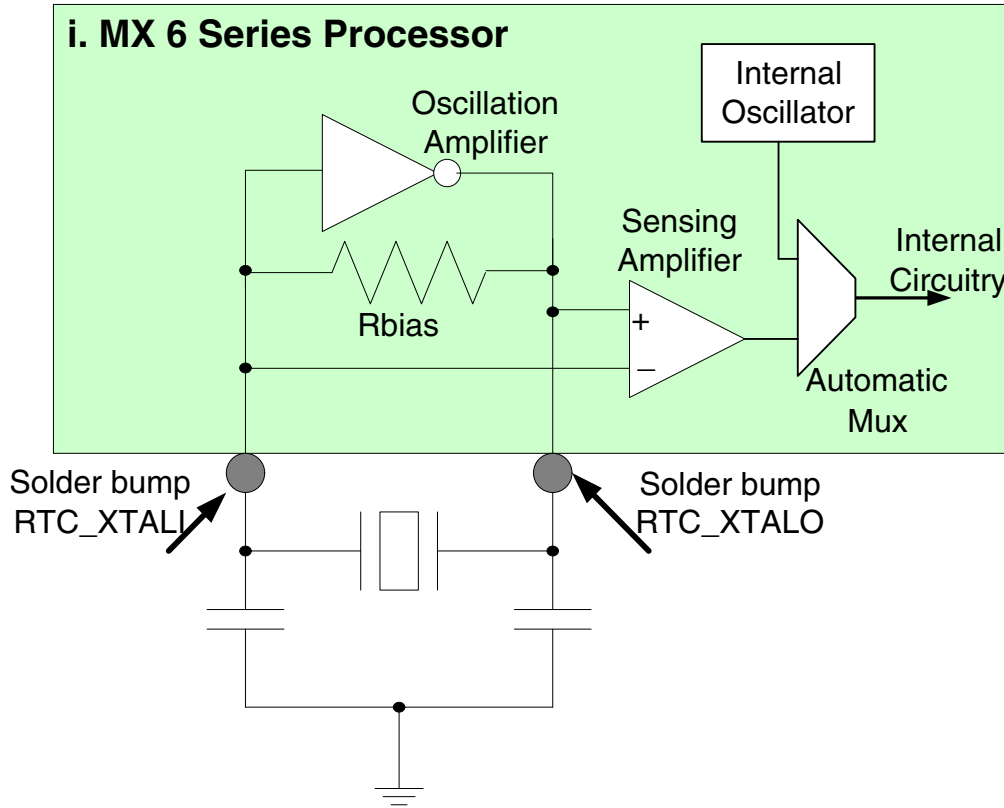
A submodule exists that gives a fairly crude (relative to the accuracy of a crystal) estimation of whether the clock frequency is correct.

This function may be enabled by setting the XTALOSC24M\_MISC0[OSC\_XTALOK\_EN] bit. It is disabled at system reset. When the oscillator is stable and the correct frequency is detected, the XTALOSC24M\_MISC0[OSC\_XTALOK] bit will be set. Note that the correct frequency will be observed before the oscillator fully blooms(the oscillation waveform build-up is completed).

## **56.4 Crystal Oscillator 32 kHz**

### 56.4.1 Oscillator Configuration (32 kHz)

The basic block diagram of the 32 kHz module configured as a crystal oscillator is shown below.



**Figure 56-2. Oscillator Configuration (32 kHz)**

This integrated biased amplifier can be used to create different frequency oscillators with different external component selection. Generally, RTC oscillators are either implemented with 32 kHz or 32.768 kHz crystals. Please consult the Security Reference Manual for appropriate frequency selection and configuration. Care must be taken to limit external leakage as this may debias the amplifier and degrade the gain.

The internal oscillator is automatically multiplexed in the clocking system when the system detects a loss of clock. The internal oscillator is not precise relative to a crystal. While it will provide a clock to the system, it generally will not be precise enough for long term time keeping. The internal oscillator is anticipated to be useful for quicker startup times, tampering prevention, and for cost savings in applications not needing the precision of an external crystal.

## 56.4.2 Bypass Configuration (32 kHz)

If it is desired to drive the chip with an external clock source, then the 32 kHz oscillator could be driven in one of three configurations using a nominal 1.1V source.

1. A single ended external clock source can be used to overdrive the output of the amplifier (RTC\_XTALO). Since the oscillation sensing amplifier is differential, the RTC\_XTALI pin should be externally floating and capacitively loaded. The combination of the internal biasing resistor and the external capacitor will filter the signal applied to the RTC\_XTALO pin and develop a rough reference for the sensing amplifier to compare to.
2. A single ended external clock source can be used to drive RTC\_XTALI. In this configuration, RTC\_XTALO should be left externally floating.
3. A differential external clock source can be used to drive both RTC\_XTALI and RTC\_XTALO.

Generally, configuration 2 is anticipated to be the most used configuration, but all three configurations may be utilized.

## 56.5 XTALOSC 24MHz Memory Map/Register Definition

### NOTE

The register content is mixed with analog functions not related to the oscillator function. These bits are noted.

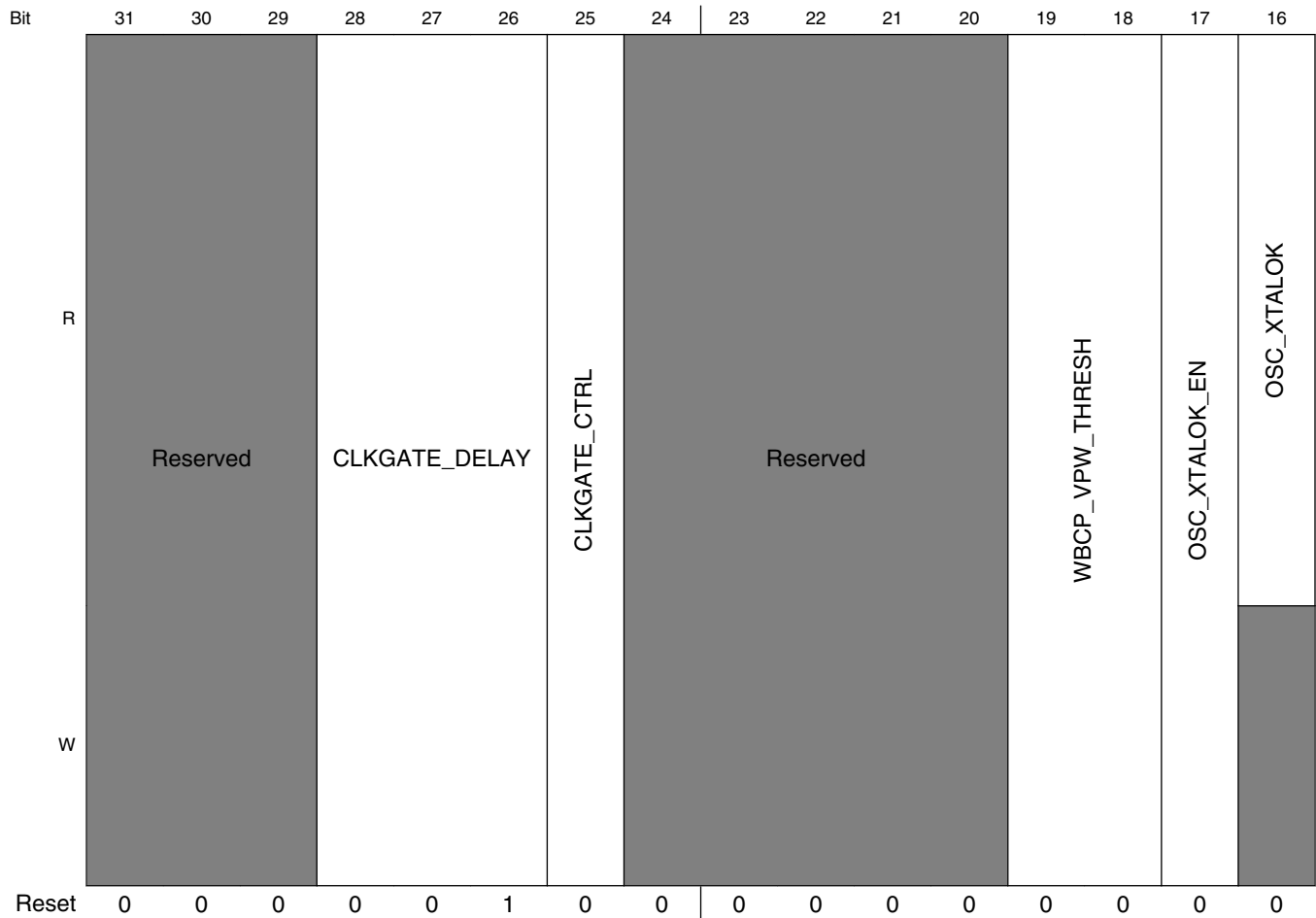
**XTALOSC24M memory map**

Absolute address (hex)	Register name	Width (in bits)	Access	Reset value	Section/ page
20C_8150	Miscellaneous Register 0 (XTALOSC24M_MISC0)	32	R/W	0400_0000h	<a href="#">56.5.1/3475</a>
20C_8260	XTAL OSC MISC1 Control Register (XTALOSC24M_MISC1)	32	R/W	0000_4009h	<a href="#">56.5.2/3479</a>
20C_8264	XTAL OSC MISC1 Control Register (XTALOSC24M_MISC1_SET)	32	R/W	0000_4009h	<a href="#">56.5.2/3479</a>
20C_8268	XTAL OSC MISC1 Control Register (XTALOSC24M_MISC1_CLR)	32	R/W	0000_4009h	<a href="#">56.5.2/3479</a>
20C_826C	XTAL OSC MISC1 Control Register (XTALOSC24M_MISC1_TOG)	32	R/W	0000_4009h	<a href="#">56.5.2/3479</a>

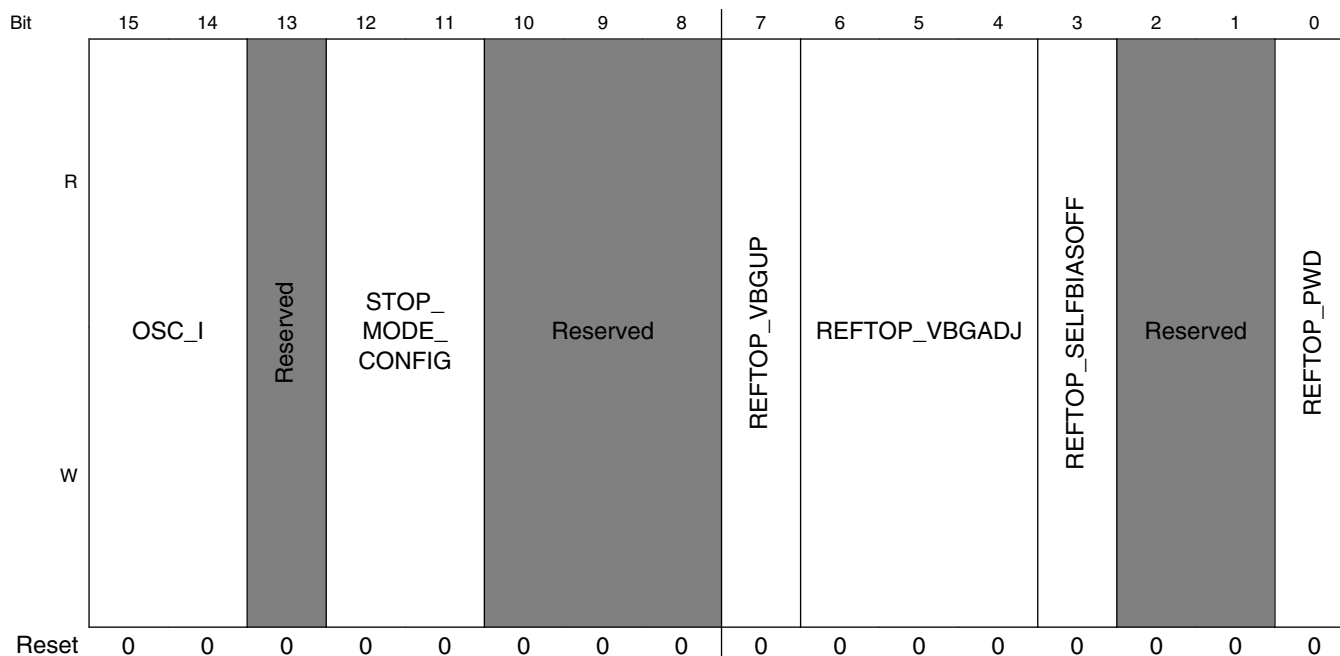
### 56.5.1 Miscellaneous Register 0 (XTALOSC24M\_MISC0)

This register defines the control and status bits for miscellaneous analog blocks.

Address: 20C\_8000h base + 150h offset = 20C\_8150h



## XTALOSC 24MHz Memory Map/Register Definition



### XTALOSC24M\_MISC0 field descriptions

Field	Description																
31–29 -	This field is reserved. Reserved																
28–26 CLKGATE_DELAY	<p>This field specifies the delay between powering up the XTAL 24MHz clock and release the clock to the digital logic inside the analog block.</p> <p><b>NOTE:</b> Do not change the field during a low power event. This is not a field that the user would normally need to modify</p> <table> <tr><td>000</td><td>0.5ms</td></tr> <tr><td>001</td><td>1.0ms</td></tr> <tr><td>010</td><td>2.0ms</td></tr> <tr><td>011</td><td>3.0ms</td></tr> <tr><td>100</td><td>4.0ms</td></tr> <tr><td>101</td><td>5.0ms</td></tr> <tr><td>110</td><td>6.0ms</td></tr> <tr><td>111</td><td>7.0ms</td></tr> </table>	000	0.5ms	001	1.0ms	010	2.0ms	011	3.0ms	100	4.0ms	101	5.0ms	110	6.0ms	111	7.0ms
000	0.5ms																
001	1.0ms																
010	2.0ms																
011	3.0ms																
100	4.0ms																
101	5.0ms																
110	6.0ms																
111	7.0ms																
25 CLKGATE_CTRL	<p>This bit allows disabling the clock gate (always un-gated) for the xtal 24MHz clock that clocks the digital logic in the analog block.</p> <p><b>NOTE:</b> Do not change the field during a low power event. This is not a field that the user would normally need to modify</p> <table> <tr><td>0</td><td>Allow the logic to automatically gate the clock when the XTAL is powered down.</td></tr> <tr><td>1</td><td>Prevent the logic from ever gating off the clock.</td></tr> </table>	0	Allow the logic to automatically gate the clock when the XTAL is powered down.	1	Prevent the logic from ever gating off the clock.												
0	Allow the logic to automatically gate the clock when the XTAL is powered down.																
1	Prevent the logic from ever gating off the clock.																
24–20 -	This field is reserved. Reserved.																

Table continues on the next page...



**XTALOSC24M\_MISC0 field descriptions (continued)**

Field	Description
	<b>NOTE:</b> These bits should always be set to zero.
19–18 WBCP_VPW_THRESH	<p>This signal alters the voltage that the pwell is charged pumped to. Default value should not be changed without guidance from Freescale</p> <p><b>NOTE:</b> Not related to oscillator.</p> <p>00 Nominal output pwell bias voltage.  01 Increase pwell output voltage by 25mV.  10 Decrease pwell output pwell voltage by 25mV.  11 Decrease pwell output pwell voltage by 50mV.</p>
17 OSC_XTALOK_EN	<p>Enable bit for the xtal_ok module(24 MHz)</p> <p>0 Xtal_ok function disabled  1 Xtal_ok function enabled</p>
16 OSC_XTALOK	<p>Status bit which signals that the output of the 24MHz crystal oscillator is stable. Generated from a timer and active detection of the actual frequency.</p> <p>0 Xtal clock not ok for use.  1 Xtal clock ok for use.</p>
15–14 OSC_I	<p>This bit field determines the bias current in the 24MHz oscillator. The idea is to start up with the highest bias current which can be decreased after startup if determined to be acceptable. Acceptable meaning that the oscillation signal amplitudes are still substantially full scale over the product operating conditions. It only makes sense to lower the current to the oscillator if the oscillator's power consumption is significant relative to the consumption of the system as a whole. If the oscillator is subsequently powered down by programming or removal of its supply, this field must be returned to the nominal value to guarantee proper startup.</p> <p>00 Nominal  01 Decrease current by 12.5%  10 Decrease current by 25.0%  11 Decrease current by 37.5%</p>
13 -	<p>This field is reserved.  Reserved.</p>
12–11 STOP_MODE_CONFIG	<p>Configure the analog behavior in stop mode.</p> <p>00 All the analog domain except the RTC is powered down on STOP mode assertion  01 All the analog domain except the LDO_1P1 and LDO_2P5 regulators are powered down on STOP mode assertion. If required the CCM can be configured to not power down the oscillator (XTALOSC)  10 Reserved  11 Reserved</p>
10–8 -	<p>This field is reserved.  Reserved</p>
7 REFTOP_VBGUP	<p>Status bit which signals that the analog bandgap voltage is up and stable.</p> <p>0 - Bandgap voltage not stable  1- Bandgap voltage stable</p> <p><b>NOTE:</b> Not related to oscillator.</p>

*Table continues on the next page...*

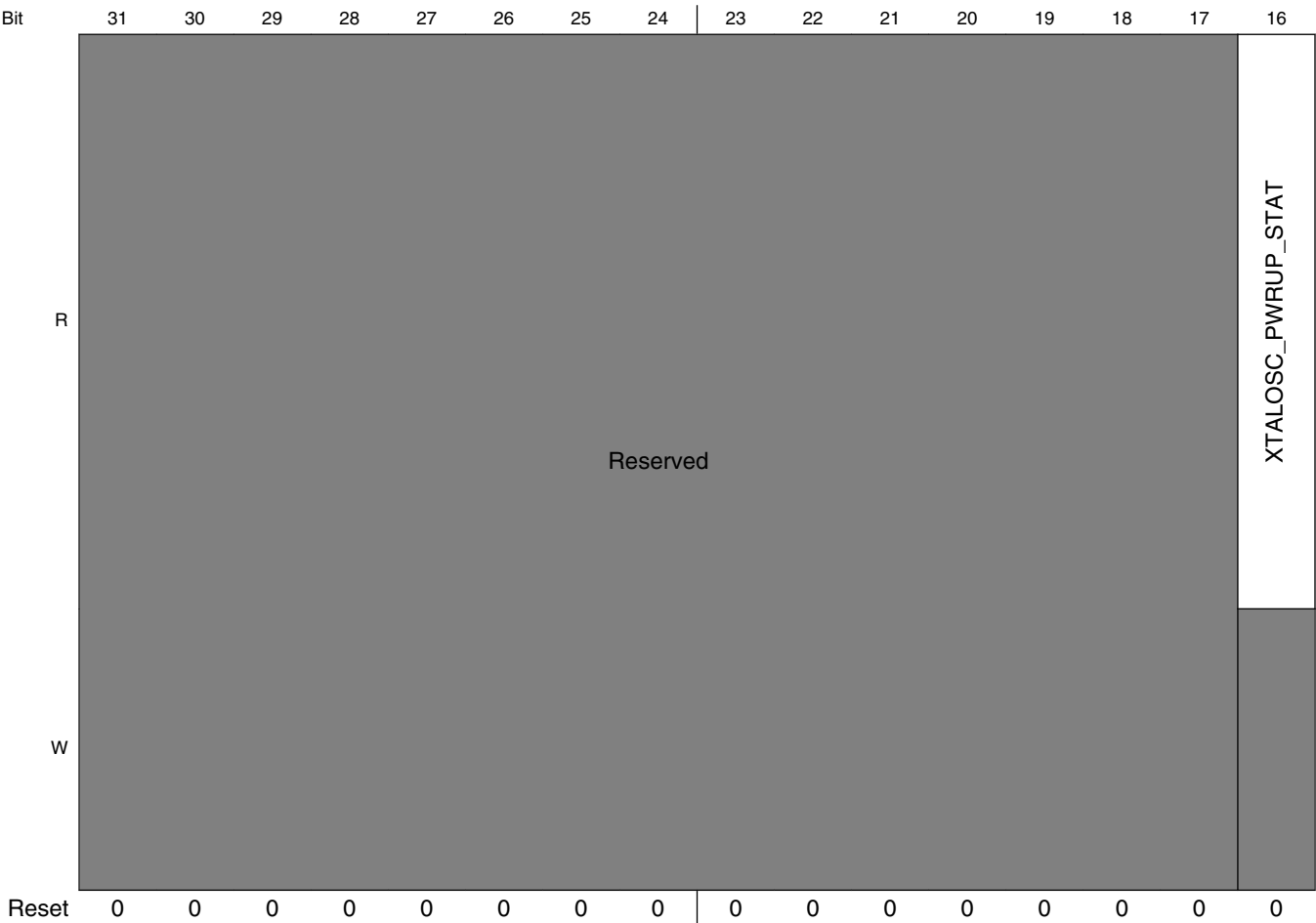
## XTALOSC24M\_MISC0 field descriptions (continued)

Field	Description
6-4 REFTOP_ VBGADJ	<p><b>NOTE:</b> Not related to oscillator.</p> <p>000 Nominal VBG  001 VBG+0.78%  010 VBG+1.56%  011 VBG+2.34%  100 VBG-0.78%  101 VBG-1.56%  110 VBG-2.34%  111 VBG-3.12%</p>
3 REFTOP_ SELFBIASOFF	<p>Control bit to disable the self-bias circuit in the analog bandgap. The self-bias circuit is used by the bandgap during startup. This bit should be set after the bandgap has stabilized and is necessary for best noise performance of analog blocks using the outputs of the bandgap.</p> <p><b>NOTE:</b> Not related to oscillator.</p> <p><b>NOTE:</b> Value should be returned to zero before removing vddhigh_in or asserting bit 0 of this register (REFTOP_PWD) to assure proper restart of the circuit.</p> <p>0 Uses coarse bias currents for startup  1 Uses bandgap based bias currents for best performance.</p>
2-1 -	<p>This field is reserved.  Reserved</p>
0 REFTOP_PWD	<p>Control bit to power-down the analog bandgap reference circuitry.</p> <p><b>NOTE:</b> Not related to oscillator.</p> <p><b>NOTE:</b> CAUTION - The bandgap reference is necessary for correct operation of most of the LDOs, PLLs, and other analog functions on the die.</p> <p>0 Bandgap reference is enabled.  1 Bandgap reference is disabled. Current consumption is removed from the supply via internal configuration.</p>

### 56.5.2 XTAL OSC MISC1 Control Register (XTALOSC24M\_MISC1*n*)

XTAL OSC miscellaneous register 1.

Address: 20C\_8000h base + 260h offset + (4d × i), where i=0d to 3d



## XTALOSC 24MHz Memory Map/Register Definition

Bit	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R	XTALOSC_PWRUP_DELAY		RCOSC.CG_OVERRIDE	RWB_EN	DISPLAY_PWRGATE	CPU_PWRGATE	L2_PWRGATE	L1_PWRGATE	REFTOP_IBIAS_OFF	LPBG_TEST	LPBG_SEL	OSC_SEL	RC_OSC_PROG			RC_OSC_EN
W																
Reset	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0	1

### XTALOSC24M\_MISC1n field descriptions

Field	Description
31–17 -	This field is reserved. Reserved.
16 XTALOSC_PWRUP_STAT	Status of the 24MHz xtal oscillator. 0 Not stable 1 Stable and ready to use
15–14 XTALOSC_PWRUP_DELAY	Specifies the time delay between when the 24MHz xtal is powered up until it is stable and ready to use. 00 0.25ms 01 0.5ms 10 1ms 11 2ms
13 RCOSC.CG_OVERRIDE	For debug purposes only. This bit effects clock gating of certain digital logic clocked by the 24MHz clk.
12 RWB_EN	Reverse well bias enable control. <b>NOTE:</b> Not related to oscillator. 0 Disabled 1 Enabled
11 DISPLAY_PWRGATE	Display logic power gate control. Used as software override. <b>NOTE:</b> Not related to oscillator.
10 CPU_PWRGATE	CPU power gate control. Used as software override.

Table continues on the next page...

**XTALOSC24M\_MISC1n field descriptions (continued)**

Field	Description
	<b>NOTE:</b> Not related to oscillator.
9 L2_PWRGATE	L2 power gate control. Used as software override. <b>NOTE:</b> Not related to oscillator.
8 L1_PWRGATE	L1 power gate control. Used as software override. <b>NOTE:</b> Not related to oscillator.
7 REFTOP_IBIAS_ OFF	Low power reftop ibias disable. <b>NOTE:</b> Not related to oscillator.
6 LPBG_TEST	Low power bandgap test bit. <b>NOTE:</b> Not related to oscillator.
5 LPBG_SEL	Bandgap select. <b>NOTE:</b> Not related to oscillator. 0 Normal power bandgap 1 Low power bandgap
4 OSC_SEL	Select the source for the 24MHz clock. 0 XTAL OSC 1 RC OSC
3–1 RC_OSC_PROG	RC osc. tuning values.
0 RC_OSC_EN	RC Osc. enable control. 0 Use XTAL OSC to source the 24MHz clock 1 Use RC OSC



# Appendix A

## SDMA Scripts

### A.1 Introduction

This appendix provides descriptions of scripts that may be used to perform data transfers using the Smart DMA (SDMA) block of the SoC.

The SDMA block supports data transfer from core memory space to core memory, from core memory space to peripherals, and vice versa.

#### A.1.1 SDMA Scripts Overview

[Table A-1](#) gives an overview of the SDMA scripts.

**Table A-1. SDMA Scripts Overview**

Data Transfer	Script to Use	Use Case	Parameters Through Context	Parameters Through Buffer Descriptor
Memory	ap_2_ap	Memory Copy	None	Word length (2'b01 or 2'b10 or 2'b11 or 2'b00), Counter, first address is memory source address, second address is memory destination address
Shared UART	uartsh_2_mcu	Uart Rx	UART Rx FIFO address (r6) Event_mask (r1) Event2_mask (r0) Watermark level (r7)	Periph size (2'b01), Counter, Memory destination address, Second address not used
UART	uart_2_mcu	Uart Rx	Uart Rx FIFO address (r6) Event_mask (r1) Event2_mask (r0) Watermark level (r7)	Periph size (2'b01), Counter, Memory source address, Second address not used
SPDIF	mcu_2_spdif	Spdif Playback	SPDIF Tx FIFO address (r6) Event_mask (r1) Event2_mask (r0) Watermark level (r7)	Periph size (2'b10 or 2'b00), Counter, Memory source address, Second address not used

*Table continues on the next page...*

**Table A-1. SDMA Scripts Overview (continued)**

Data Transfer	Script to Use	Use Case	Parameters Through Context	Parameters Through Buffer Descriptor
	spdif_2_mcu	Spdif Record	SPDIF Rx FIFO address (r6) Event_mask (r1) Event2_mask (r0) Watermark level (r7)	Periph size (2'b10 or 2'b00), Counter, Memory source address, Second address not used
Shared Peripheral	mcu_2_shp	Peripheral Transmit	Tx fifo address (r6) Event_mask (r1) Event2_mask(r0) Watermark level (r7)	Periph size (2'b01 or 2'b10 or 2'b11 or 2'b00), Counter, Memory source address, Second address not used
	shp_2_mcu	Peripheral Receive	RX fifo address (r6) Event_mask (r1) Event2_mask(r0) Watermark level (r7)	Periph size (2'b01 or 2'b10 or 2'b11 or 2'b00), Counter, Memory destination address, Second address not used
Peripheral	mcu_2_app	Peripheral Transmit	Tx fifo address (r6) Event_mask (r1) Event2_mask(r0) Watermark level (r7)	Periph size (2'b01 or 2'b10 or 2'b11 or 2'b00), Counter, Memory source address, Second address not used
	app_2_mcu	Peripheral Receive	RX fifo address (r6) Event_mask (r1) Event2_mask(r0) Watermark level (r7)	Periph size (2'b01 or 2'b10 or 2'b11 or 2'b00), Counter, Memory destination address, Second address not used
SSI	mcu_2_ssiapp	Audio Playback	Tx FIFO 0 address (r6) Event_mask (r1) Event2_mask (r0) Watermark level (r7)	Periph size (2'b01 or 2'b10 or 2'b11 or 2'b00), Counter, Memory source address, Second address not used
	ssiapp_2_mcu	Audio Record	Rx FIFO 0 address (r6) Event_mask (r1) Event2_mask (r0) Watermark level (r7)	Periph size (2'b01 or 2'b10 or 2'b11 or 2'b00), Counter, Memory destination address, Second address not used
Shared SSI	mcu_2_ssish	Audio Playback	Tx FIFO 0 address (r6) Event_mask (r1) Event2_mask (r0) Watermark level (r7)	Periph size (2'b01 or 2'b10 or 2'b11 or 2'b00), Counter, Memory source address, Second address not used
	ssish_2_mcu	Audio Record	Rx FIFO 0 address (r6) Event_mask (r1) Event2_mask (r0) Watermark level (r7)	Periph size (2'b01 or 2'b10 or 2'b11 or 2'b00), Counter, Mem destination address, Second address not used
Peripheral	p_2_p	Peripheral Transfer	Destination address (r6) Event_mask (r1) Event2_mask (r0) Source address (r2) Information (r7)	Counter, First address and Second address not used
HDMI	hdmi_dma	HDMI playback	Buffer chain length (r4) Pointer address (r6)	None

In [Table A-1](#), the data transfer column lists the possible types of DMA channels that involve the SDMA and a specific script is attached to each channel. It must be noted that some scripts cover several DMA channels. The scripts deal with the channels that have generic peripherals where only the watermark level is needed (no aging timer, no end\_of\_packet feature, and so on). The location refers to whether a script resides in ROM or in RAM and the size of the script is given in the instructions. The parameters columns are explained in the next section.



The following terms are used in [Table A-1](#) which lists all the supported data transfers.

**EMI or External Memory:**

The external memories are connected to the External Memory Interface. Thus, a data transfer to EMI means a data transfer to any of the external memories (NAND Flash, NOR Flash, PSRAM, SDRAM, and so on).

**Host mem or ARM internal memory:**

The memory space accessible through the MAX of the ARM platform. It does not correspond to the peripherals, although they are accessible through the MAX as well. A data transfer from Host mem means data is read from the internal memory of the ARM or from the external memory. It is possible to point to an external memory through the MAX because it is also connected to EMI module. The next diagram replaces the SDMA and its DMA port in the hardware architecture. The Host mem is accessible through the Per DMA port of the SDMA, the EMI is accessible through the Burst DMA and the DSP mem through the DSP port.

**Shared Peripheral:**

A same peripheral can be connected to the shared peripheral bus (output of SPBA module) and to the ARM platform. This is the case for some UART, CSPI, and SSI. Therefore, "shared UART" indicates that the UART is connected to the shared peripheral, whereas "UART" means the UART is connected to the ARM platform.

[Figure A-1](#) gives an overview of the SDMA in its hardware environment.

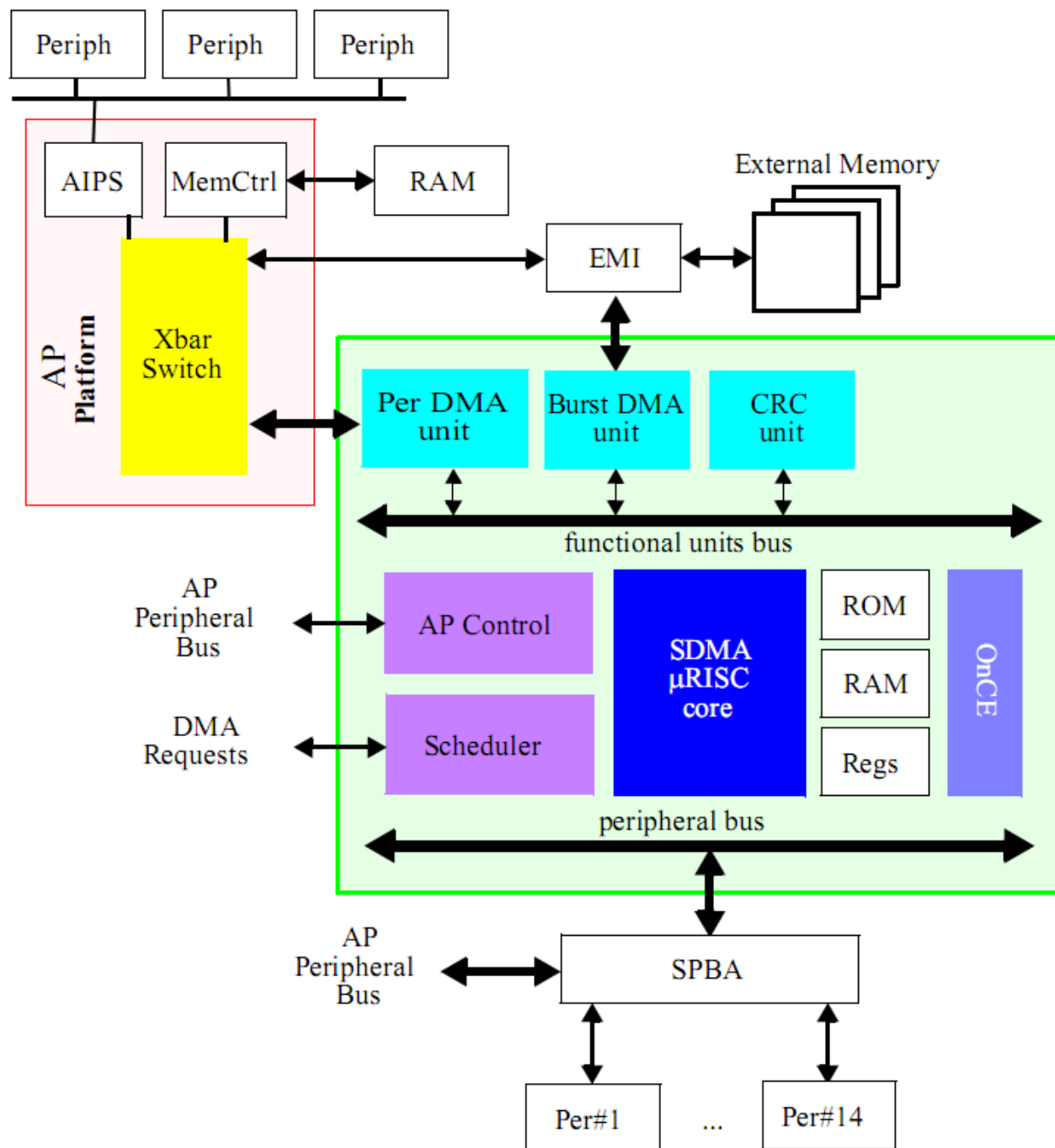


Figure A-1. SDMA Connections

## A.2 Scripts Parameters: Definition

## A.2.1 Parameters Definition

### A.2.1.1 Parameters Required by Data Transfer Script Involving a Peripheral

The scripts that have a peripheral as a player in the transfer of data need to have the following input parameters:

#### A.2.1.1.1 Watermark Level (WML)

It is the upper or the lower threshold of the receive or transmit FIFO of the peripheral that triggers a DMA request to the SDMA.

The WML determines the data transfer loop size, meaning the number of bytes that will be read/write from/to the receive/transmit FIFO. This parameter must be a multiple of the peripheral FIFO data size. As an example, if UART1 is the data transfer destination, data must be written to the Tx FIFO of the peripheral. For instance, the watermark level can be set to 8 bytes (the UART is a 1 byte FIFO data size), meaning the UART\_TX DMA request will be sent to the SDMA each time there are more than 8 free locations in the FIFO; therefore SDMA loop size will be set to 8 and 8 bytes will be written to the UART\_TX FIFO. In fact, the loop size is the minimum between the WML and the count field of the buffer descriptor attached to the channel. This concept is explained later on in the chapter. Similarly, if the UART is the source of a data transfer and if the WML is set to 16, each time the DMA request is received by the SDMA, which triggers the associated channel, the data transfer loop size is set to 16. This means 16 bytes will be read from the UART\_RX FIFO. The watermark level is a "long" (32-bit data) programmed by the ARM and is not supposed to evaluate a lot during application. The WML is always given in bytes.

#### NOTE

For the transmit FIFO, the  $WML = FIFO\_SIZE - FIFOTX\_THRESHOLD$  For the receive FIFO, the  $WML = FIFORX\_THRESHOLD$  The  $FIFOTX\_THRESHOLD / FIFORX\_THRESHOLD$  are the values of the peripheral transmit and receive FIFO level at which a DMA request is triggered. For instance, if TX FIFO is 32 bytes depth and if the DMA request is sent when the number of bytes present in the FIFO is below a threshold of 10, the WML will be 22. It means that when the SDMA will receive the DMA request from the

peripheral, it will be possible to store 22 bytes in the TX FIFO.  
For the receive FIFO, the WML equals the threshold.

#### **A.2.1.1.2 Event\_mask and Event2\_mask**

As previously said, when the WML threshold is over or under passed, a DMA request is sent to the scheduler part of the SDMA.

The SDMA scheduler has 48 input pins, called "events", which are connected to the different DMA request.

The mapping between the DMA request name and the event number is SoC dependent and defined in the "Interrupts and SDMA Events" chapter. For instance, in SoC A, the UART\_RX DMA request is connected to events pin 5; whereas for SoC B, it may be connected to events pin 15. The script reads the EVENTS(32 events requests) and EVENTS2(remaining 16 events requests) register to check if the received DMA request is compliant with the expected one and it guaranteed that a second DMA request sent during the data transfer is not lost.

For instance, while SDMA is reading the UART\_RX FIFO (again, the number of data to be read is given by WML value), the peripheral may receive new data from its input ports. Therefore after the first data transfer, the number of data available in the UART received FIFO can be still higher than the WML threshold, so a second DMA request may be sent.

The event\_mask value is generally a 1-bit high value, which means that if the script attached to the channel must be triggered by DMA request number I, event\_mask[I] must be set to 1, if the script attached to the channel must be triggered by DMA request number 32+I, event2\_mask[I] must be set to 1. Some scripts (like ata\_2\_mcu) may be triggered by several DMA requests; in that case several bits of the event\_mask must be asserted. The event\_mask and event2\_mask are long (32-bit) data.

#### **A.2.1.1.3 Peripheral Address**

The scripts of the SDMA scripts library are SoC independent but as their goal is to address peripheral, they required the base address or the FIFO address of the peripheral.

Passing FIFO or peripheral base address depends on the scripts.

#### **A.2.1.1.4 Data Length**

The scripts whose name contains the key word "app" or "shp" are generic: they are used to transfer data from/to a 8, 16, 24 or 32-bit peripheral data size.

Some jumps to some routines of these scripts depend on this peripheral size. This parameter is required as input of these scripts. This parameter is passed through the command field of the buffer descriptor and is coded on bits 25 and 24 of the mode:

00: 32-bit data transfer.

01: 8-bit data transfer.

10: 16-bit data transfer.

11: 24-bit data transfer.

## A.3 Scripts

The following scripts are all based on buffer descriptor mechanism.

### A.3.1 SDMA ROM Scripts

The ROM holds the generic memory to memory scripts (ap\_2\_ap, ap\_2\_bp, bp\_2\_ap, bp\_2\_bp) and the peripheral most useful scripts (app\_2\_mcu, mcu\_2\_app, uart\_2\_mcu, uartsh\_2\_mcu, mcu\_2\_shp, shp\_2\_mcu).

In these memory to memory scripts, the number of bytes to transfer from the source memory to the destination is divided by 4 to get the number of 32-bit words that can be transferred. Most of the time, the transfer will be a transfer of words. There will be a byte or half data transfer at the end of the transfer if the total number of bytes to transfer is not an multiple of 4.

#### A.3.1.1 ap\_2\_ap

This script is used to transfer data inside the application processor memory using BurstDMA.

Using this script there is no restriction on the number of bytes to transmit, the source address and destination address alignment, or on the source memory and destination memory type. But word alignment and source/destination memory type will impact performance. In case that the source and destination are in the same memory type, being word aligned allows the SDMA to use the copy capability of the DMA units. When it is not the case the SRAM of the SDMA is used as temporary buffer.

### **A.3.1.1.1 Parameters Transmitted Through the Context ap\_2\_ap**

#### **A.3.1.1.2 Parameters Transmitted Through the Buffer Descriptor ap\_2\_ap**

The number of bytes to transmit is stored in the first Buffer descriptor word.

The source address is stored in the second Buffer descriptor word (address field).

The destination address is stored in the Extended Buffer address

#### **A.3.1.1.3 Use of the General Register During the Execution ap\_2\_ap**

- r0: nb of bytes to transfer, counter during loop
- r2: channel cb address, scratch
- r3: current buffer descriptor pointer, AP destination address, scratch
- r4: mode
- r5: source address loaded by getbd,
- r6: destination address loaded by getcb, scratch
- r7: pointer in the scratch buffer

#### **A.3.1.1.4 Overview of Script Functionality ap\_2\_ap**

The parameters of the transfer (number of bytes to transfer, source address and destination address) are retrieved from the buffer descriptor.

Depending on the address alignment and the DMA units involved, if possible, the transfer is performed using the copy capability of the DMA unit, or using the SDMA SRAM as a temporary buffer.

When the transfer is finished, this algorithm restarts (a new buffer descriptor is read).

The count in the BD mode word after the script is over is meaningful only if error was reported, else it has the same value as was fed as input to the script.

### **A.3.1.2 uartsh\_2\_mcu**

This script is used to transfer data from the shared UART to the External memory. The UART is connected to the ARM platform.

The UART is an 8-bit peripheral, but when reading the FIFO, 16 bits are retrieved. In fact, the 8 MSB are the status bytes for the current data. At each access the value of this byte is checked to verify that a valid data was retrieved. If an error is detected the transfer is stopped and the information is sent back to the Host through the buffer descriptor with the byte count field updated.

#### **A.3.1.2.1 Parameters Transmitted Through the Context `uartsh_2_mcu`**

r0: mask to check events2 - If script is triggered by event 32+I, r0[I] must be set to 1. (Project dependent)

r1: mask to check event - If script is triggered by event I, r1[I] must be set to 1. (Project dependent)

r6: address of the peripheral Rx FIFO (project dependent)

r7: Watermark level - Used to determine the maximum of data that can be sent to the peripheral each time the channel is started.

#### **A.3.1.2.2 Parameters Transmitted Through the Buffer Descriptor `uartsh_2_mcu`**

The number of bytes to transmit is stored in the first Buffer descriptor word.

The destination address is stored in the second Buffer descriptor word (address field).

The Extended Buffer address is not used.

#### **A.3.1.2.3 Use of the General Register During the Execution `uartsh_2_mcu`**

r0 = mask to check events2

r1 = mask to check events

r2 = AP M3 destination address

r3 = Ram channel context address

r4 = Bd mode

r5 = Bd mode Count

r6 = SDMA memory mapped regs base address

r7 = Watermark

#### A.3.1.2.4 Overview of Script Functionality `uartsh_2_mcu`

When the UART sends a DMA request, it can be for three different reasons.

When the RX threshold is over passed. It deals with the most frequent event.

When the AGEING timer has reached its final value, meaning there is less data than the RX threshold in the RX FIFO, and they have been present for too long. It is called the aging DMA request.

When IDLE timer reached its final value; meaning the RX FIFO has been empty for too long. It is called the iddle dma request. The IDLE timer is not used, so only normal and AGEING DMA requests are supported by the script.

The first time the script is executed, a buffer descriptor is opened, the buffer destination address and its size are retrieved from the BD.

Then the script checks if the RRDY bit of USR1 is set. If yes, it means that the Watermark level has been reached, there are "WML" bytes in RX FIFO. If the count field of the opened BD is higher than WML-1, the SDMA script will perform WML-1 read accesses to the RX FIFO and WML write access to the destination buffer (a read access then a write access is call a data transfer loop). If the count field is lesser than the WML-1 value, the count field will be the data transfer loop size. So the data transfer loop size equals  $\min(\text{WML}-1, \text{BD count})$

So there will be always one data remaining in the UART RX FIFO after every data transfer loop and the ageing timer will always trigger a DMA request. It means that the channel on which the script is run will be always closed on an AGEING DMA request.

Now assume the script is triggered by a DMA request, but RRDY is not set. It means that it deals with an AGEING DMA request; there is at least one data in the RX FIFO. The data is read and written into the destination buffer, and then the RDR is checked again. If it is set and if the destination data buffer is not full (BD count is not 0), a new data is read and so on until RDR is 0. When RDR is 0, the ageing timer interrupt flag is disabled by setting the AGTIM bit, the count field of BD is updated and the current BD is closed. Then the script tries to open the next BD if the Continuous bit of the current BD was set, if there is no more BD to open, the channel is stopped.

When reading a data from the UART RX FIFO, the MSB are check to verify that a valid data has been retrieved. If an error is detected the transfer is stopped, the error bit is set in the BD, the count is updated, the BD is closed and an interrupt is sent to the ARM.

When the buffer is full, this algorithm restarts (a new buffer descriptor is read, if it exists).



The next diagram illustrates the script algorithm, it is quite complex but things become more obvious when we notice that there is a loop for the normal DMA request (in sky-blue) and on when the AGEING DMA request was detected (in yellow).

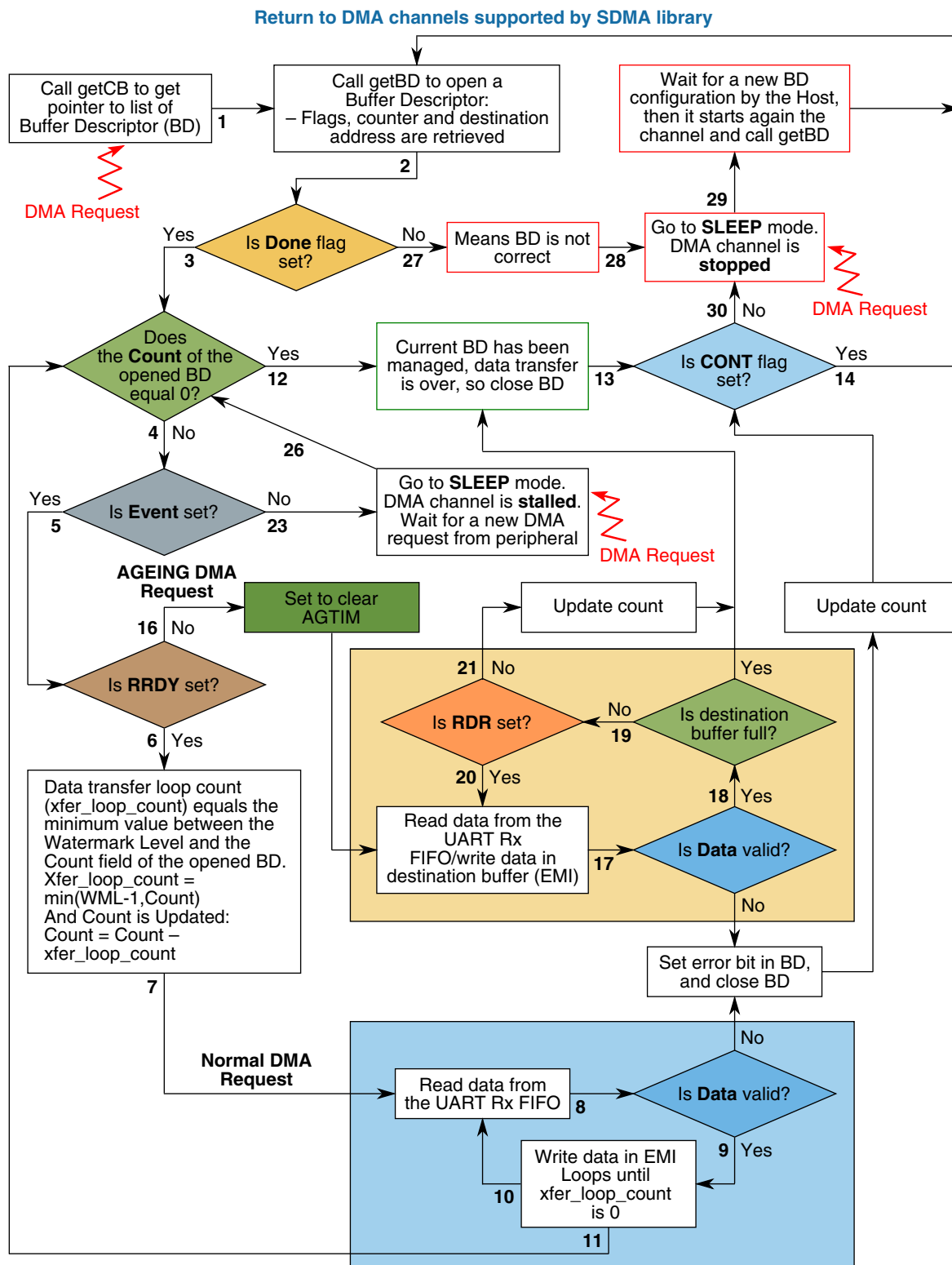


Figure A-2. Script Algorithm

### **A.3.1.3 shp\_2\_mcu**

This generic script is used to transfer data from a 8, 16, 24 or 32-bit peripheral connected to the shared peripheral bus accessed through the SPBA to memories accessed by the BurstDMA (External memories).

It can be used for SSI (8, 16, 24 or 32-bit data size), for CSPI (32-bit data size) or for SDHC(MMC)/SIM (16-bit data size), these peripherals being connected to the shared peripheral bus.

#### **A.3.1.3.1 Parameters Transmitted Through the Context shp\_2\_mcu**

r0: mask to check events2 - If script is triggered by event 32+I, r0[I] must be set to 1. (Project dependent)

r1: mask to check event - If script is triggered by event I, r1[I] must be set to 1. (Project dependent)

r6: address of the peripheral Rx FIFO (project dependent)

r7: Watermark level - Used to determine the maximum of data that can be sent to the peripheral each time the channel is started.

#### **A.3.1.3.2 Parameters Transmitted Through the Buffer Descriptor shp\_2\_mcu**

The peripheral size/data length is set in the command field of the first Buffer descriptor word, especially bits 24 and 25.

The number of bytes to transmit is stored in the first Buffer descriptor word (count field)

The destination address in the external memory is stored in the second Buffer descriptor word (address field).

The Extended Buffer address is not used.

#### **A.3.1.3.3 Use of the General Register During the Execution shp\_2\_mcu**

r0 = mask to check events2

r1 = mask to check events

r2 = BD destination address

r3 = Ram channel context address  
 r4 = Bd mode  
 r5 = Bd mode Count  
 r6 = SDMA memory mapped regs base address  
 r7 = Watermark

#### **A.3.1.3.4 Overview of Script Functionality shp\_2\_mcu**

The parameters of the transfer (peripheral size, number of bytes to transfer and source address) are retrieved from the buffer descriptor.

When the peripheral sends an event, a data transfer loop is executed. This event is set when there is at least "watermark level" data in the FIFO.

During this data transfer loop, the number of bytes that are transferred from the RX FIFO of the peripheral to the EMI is equal to the minimum value between the "watermark level" and the number of data that have not been transferred yet. For each data transfer (one read, one write), the size of the memory write access is fixed by the parameter peripheral size.

The necessary number of transfer loops are executed in order to fill the buffer, but they are executed only if the event is set. While the event is not set, the channel is not executable.

When the buffer is filled, this algorithm restarts (a new buffer descriptor is read).

#### **A.3.1.4 mcu\_2\_shp**

This generic script is used to transfer data from memories accessed by the BurstDMA (External memories) to a 8, 16, 24 or 32-bit peripheral connected to the shared peripheral bus accessed through the SPBA

. It can be used for SSI (8, 16, 24 or 32-bit data size), for CSPI (32-bit data size), for SDHC(MMC)/SIM (16-bit data size) or for UART3,4 (8-bit data size), these peripherals being connected to the shared peripheral bus.

##### **A.3.1.4.1 Parameters Transmitted Through the Context mcu\_2\_shp**

r0: mask to check events2 - If script is triggered by event 32+I, r0[I] must be set to 1.  
 (Project dependent)

r1: mask to check events - If script is triggered by event I, r1[I] must be set to 1. (Project dependent)

r6: address of the peripheral Tx FIFO (project dependent)

r7: Watermark level - Used to determine the maximum of data that can be retrieved from the peripheral each time the channel is started.

#### **A.3.1.4.2 Parameters Transmitted Through the Buffer Descriptor mcu\_2\_shp**

The peripheral size/data length is set in the command field of the first Buffer descriptor word, especially bits 24 and 25.

The number of bytes to transmit is stored in the first Buffer descriptor word (count field)

The source address in the external memory is stored in the second Buffer descriptor word (address field).

The Extended Buffer address is not used.

#### **A.3.1.4.3 Use of the General Register During the Execution mcu\_2\_shp**

r0 = mask to check events2

r1 = mask to check events

r2 = EVENTS / EVENTS2

r3 = Ram channel context address

r4 = Bd mode

r5 = Bd mode Count

r6 = SDMA memory mapped regs base address

r7 = Watermark

#### **A.3.1.4.4 Overview of Script Functionality mcu\_2\_shp**

The parameters of the transfer (peripheral size, number of bytes to transfer and source address) are retrieved from the buffer descriptor.

When the peripheral sends an event, a data transfer loop is executed. This event is set when there is at least "watermark level" empty rooms in the FIFO.

During this data transfer loop, the number of bytes that are transferred from the EMI to the TX FIFO of the peripheral is equal to the minimum value between the "watermark level" and the number of data that have not been transferred yet. For each data transfer (one read, one write), the size of the memory read access is fixed by the parameter peripheral size.

The necessary number of transfer loops are executed in order to empty the buffer, but they are executed only if the event is set. While the event is not set, the channel is not executable.

When the buffer is empty, this algorithm restarts (a new buffer descriptor is read).

### **A.3.1.5 uart\_2\_mcu**

This script is similar to uartsh\_2\_mcu script. The only difference between these scripts is the way SDMA access UART RX FIFO.

In case of uartsh\_2\_mcu, the UART RX FIFO can directly be accessed by SDMA as it is on Shared Peripheral Bus. In case of uart\_2\_mcu, SDMA script uses one of its DMA (Peripheral DMA) to access UART RX FIFO.

### **A.3.1.6 app\_2\_mcu**

This generic script is used to transfer data from a 8, 16, 24 or 32-bit peripheral connected to the AIPS accessed through the Peripherals DMA of SDMA, to memories accessed by the BurstDMA (External memories).

It can be used for SSI (8, 16, 24 or 32-bit data size), for CSPI (32-bit data size) or for SDHC(MMC)/SIM (16-bit data size).

#### **A.3.1.6.1 Parameters Transmitted Through the Context app\_2\_mcu**

r0: mask to check events2 - If script is triggered by event 32+I, r0[I] must be set to 1. (Project dependent)

r1: mask to check event - If script is triggered by event I, r1[I] must be set to 1. (Project dependent)

r6: address of the peripheral Rx FIFO (project dependent)

r7: Watermark level - Used to determine the maximum of data that can be sent to the peripheral each time the channel is started.

### **A.3.1.6.2 Parameters Transmitted Through the Buffer Descriptor app\_2\_mcu**

The [Data Length](#) peripheral size/data length is set in the command field of the first buffer descriptor word, specially bits 24 and 25.

The number of bytes to transmit is stored in the first buffer descriptor word (count field).

The destination address in the external memory is stored in the second buffer descriptor word (address field).

The Extended Buffer address is not used.

### **A.3.1.6.3 Use of the General Register During the Execution app\_2\_mcu**

r0 = mask to check events2

r1 = mask to check events

r2 = AP M3 destination address

r3 = Ram channel context address

r4 = Bd mode

r5 = Bd mode Count

r6 = SDMA memory mapped regs base address

r7 = Watermark

### **A.3.1.6.4 Overview of Script Functionality app\_2\_mcu**

The parameters of the transfer (peripheral size, number of bytes to transfer and source address) are retrieved from the buffer descriptor.

When the peripheral sends an event, a data transfer loop is executed. This event is set when there is at least "watermark level" data in the FIFO.

During this data transfer loop, the number of bytes that are transferred from the RX FIFO of the peripheral to the EMI is equal to the minimum value between the "watermark level" and the number of data that have not been transferred yet. For each data transfer (one read, one write), the size of the memory write access is fixed by the parameter peripheral size.

The necessary number of transfer loops are executed in order to fill the buffer, but they are executed only if the event is set. While the event is not set, the channel is not executable.

When the buffer is filled, this algorithm restarts (a new buffer descriptor is read).

### **A.3.1.7 mcu\_2\_app**

This generic script is used to transfer data from memories accessed by the BurstDMA (External memories), to a 8, 16, 24 or 32-bit peripheral connected to the AIPS accessed through the Periphra DMA of SDMA.

It can be used for SSI (8, 16, 24 or 32-bit data size), for CSPI (32-bit data size) or for SDHC(MMC)/SIM (16-bit data size).

#### **A.3.1.7.1 Parameters Transmitted Through the Context mcu\_2\_app**

r0: mask to check events2 - If script is triggered by event 32+I, r0[I] must be set to 1. (Project dependent)

r1: mask to check event - If script is triggered by event I, r1[I] must be set to 1. (Project dependent)

r6: address of the peripheral Tx FIFO (project dependent)

r7: Watermark level - Used to determine the maximum of data that can be sent to the peripheral each time the channel is started.

#### **A.3.1.7.2 Parameters Transmitted Through the Buffer Descriptor mcu\_2\_app**

The peripheral size/data length is set in the command field of the first buffer descriptor word, specially bits 24 and 25.

The number of bytes to transmit is stored in the first buffer descriptor word (count field).

The source address in the external memory is stored in the second buffer descriptor word (address field).

The Extended Buffer address is not used.

#### **A.3.1.7.3 Use of the General Register During the Execution mcu\_2\_app**

r0 = mask to check events2

r1 = mask to check events

r2 = TX FIFO address

r3 = Ram channel context address

r4 = Bd mode

r5 = Bd mode Count

r6 = SDMA memory mapped regs base address

r7 = Watermark

#### **A.3.1.7.4 Overview of Script Functionality mcu\_2\_app**

The parameters of the transfer (peripheral size, number of bytes to transfer and source address) are retrieved from the buffer descriptor.

When the peripheral sends an event, a data transfer loop is executed. This event is set when there is at least "watermark level" empty room in the FIFO.

During this data transfer loop, the number of bytes that are transferred from the EMI to the TX FIFO of the peripheral is equal to the minimum value between the "watermark level" and the number of data that have not been transferred yet. For each data transfer (one read, one write), the size of the memory read access is fixed by the parameter peripheral size.

The necessary number of transfer loops are executed in order to fill the buffer, but they are executed only if the event is set. While the event is not set, the channel is not executable.

When the buffer is filled, this algorithm restarts (a new buffer descriptor is read).

#### **A.3.1.8 mcu\_2\_spdif**

This script performs a data move from AP buffer to 24 bits FIFO of the SPDIF.

This peripheral is on the peripheral bus, the SDMA accesses it through the SPBA. The MCU buffer is accessed through the burst DMA.

##### **A.3.1.8.1 Parameters Transmitted Through the Context mcu\_2\_spdif**

r0: mask to check events  
2 - If script is triggered by event 32+I, r0[I] must be set to 1.  
(Project dependent)



r1: mask to check events - If script is triggered by event I, r1[I] must be set to 1. (Project dependent)

r6: Address of SPDIF STL register

r7: Watermark level - Used to determine the maximum data that can be sent to the peripheral each time the channel is started.

#### **A.3.1.8.2 Parameters Transmitted Through the Buffer Descriptor mcu\_2\_spdif**

The data length (Either 32-bit or 16-bit) is set in the command field of the first Buffer descriptor word, especially bits 24, 25.

The number of bytes to transmit is stored in the first Buffer descriptor word (count field)

The source address in the external memory is stored in the second Buffer descriptor word (address field).

The Extended Buffer address is not used.

#### **A.3.1.8.3 Use of the General Register During the Execution mcu\_2\_spdif**

r0 = mask to check events2

r1 = mask to check events

r2 = EVENTS / EVENTS2

r3 = Ram channel context address

r4 = Bd mode

r5 = Bd mode Count

r6 = SDMA memory mapped regs base address

r7 = Watermark

#### **A.3.1.8.4 Overview of Script Functionality mcu\_2\_spdif**

The parameters of the transfer (data length, number of bytes to transfer and source address) are retrieved from the buffer descriptor.

When the spdif sends an event, a data transfer loop is executed. This event is set when there is at least "watermark level" of empty room in the FIFO (Sum of left and right FIFO's).

During this data transfer loop, the number of bytes that are transferred from the EMI to the TX FIFO of the spdif is equal to the minimum value between the "watermark level" and the number of data that have not been transferred yet. For each data transfer (one read, one write), the size of the memory read access is fixed by the parameter data length.

The necessary numbers of transfer loops are executed in order to empty the buffer, but they are executed only if the event is set. While the event is not set, the channel is not executable.

When the buffer is empty, this algorithm restarts (a new buffer descriptor is read).

### **A.3.1.9 spdif\_2\_mcu**

This script performs a data move from 24 bits FIFO of the SPDIF to AP buffer.

This peripheral is on the peripheral bus, the SDMA accesses it through the SPBA. The MCU buffer is accessed through the burst DMA.

#### **A.3.1.9.1 Parameters Transmitted Through the Context spdif\_2\_mcu**

r0: mask to check events2 - If script is triggered by event 32+I, r0[I] must be set to 1. (Project dependent)

r1: mask to check events - If script is triggered by event I, r1[I] must be set to 1. (Project dependent)

r6: Address of SPDIF SRL register

r7: Watermark level - Used to determine the maximum data that can be sent to the peripheral each time the channel is started.

#### **A.3.1.9.2 Parameters Transmitted Through the Buffer Descriptor spdif\_2\_mcu**

The data length (Either 32 bit or 16 bit) is set in the command field of the first Buffer descriptor word, especially bits 24, 25.

The number of bytes to transmit is stored in the first Buffer descriptor word (count field)

The destination address in the external memory is stored in the second Buffer descriptor word (address field).

The Extended Buffer address is not used.

### A.3.1.9.3 Use of the General Register During the Execution `spdif_2_mcu`

r0 = mask to check events2

r1 = mask to check events

r2 = EVENTS / EVENTS2

r3 = Ram channel context address

r4 = Bd mode

r5 = Bd mode Count

r6 = SDMA memory mapped regs base address

r7 = Watermark

### A.3.1.9.4 Overview of Script Functionality `spdif_2_mcu`

The parameters of the transfer (data length, number of bytes to transfer and destination address) are retrieved from the buffer descriptor.

When the `spdif` sends an event, a data transfer loop is executed. This event is set when there is at least "watermark level" data in the FIFO (Sum of left and right FIFO's).

During this data transfer loop, the number of bytes that are transferred from the RX FIFO of the `spdif` to the EMI is equal to the minimum value between the "watermark level" and the number of data that have not been transferred yet. For each data transfer (one read, one write), the size of the memory write access is fixed by the parameter data length.

The necessary numbers of transfer loops are executed in order to empty the buffer, but they are executed only if the event is set. While the event is not set, the channel is not executable.

When the buffer is empty, this algorithm restarts (a new buffer descriptor is read).

## A.3.2 SDMA RAM scripts

### A.3.2.1 `mcu_2_ssiapp`

This script performs a data move from AP buffer to 8, 16, 24 or 32-bit FIFO of the SSI in the AP region with 2 FIFOs enabled.

The MCU buffer is accessed through the burst DMA, while SSI is accessed with Per DMA.

#### **A.3.2.1.1 Parameters Transmitted Through the Context mcu\_2\_ssiapp**

r0: mask to check events2 - If script is triggered by event 32+I, r0[I] must be set to 1. (Project dependent)

r1: mask to check events - If script is triggered by event I, r1[I] must be set to 1. (Project dependent)

r6: Address of SSI Transmit FIFO 0 register

r7: Watermark level - Used to determine the maximum data that can be sent to the peripheral each time the channel is started.

#### **A.3.2.1.2 Parameters Transmitted Through the Buffer Descriptor mcu\_2\_ssiapp**

The data length (8, 16, 24 or 32-bit) is set in the command field of the first buffer descriptor word, especially bits 24 and 25.

The number of bytes to transmit is stored in the first buffer descriptor word (count field).

The source address in the memory is stored in the second buffer descriptor word (address field).

The Extended Buffer address is not used.

#### **A.3.2.1.3 Use of the General Register During the Execution mcu\_2\_ssiapp**

r0 = mask to check events2

r1 = mask to check events

r2 = Tx FIFO address

r3 = Ram channel context address

r4 = Bd mode

r5 = Bd mode Count

r6 = SDMA memory mapped regs base address

r7 = Watermark

#### **A.3.2.1.4 Overview of Script Functionality mcu\_2\_ssiapp**

The parameters of the transfer (data length, number of bytes to transfer and source address) are retrieved from the buffer descriptor.

When the SSI sends an event, a data transfer loop is executed. This event is set when there is at least "watermark level" of empty room in the FIFO (Sum of left and right FIFO's).

During this data transfer loop, the number of bytes that are transferred from the EMI to the TX FIFO of the SSI is equal to the minimum value between the "watermark level" and the number of data that have not been transferred yet. For each data transfer (one read, one write), the size of the memory read access is fixed by the parameter data length.

The necessary numbers of transfer loops are executed in order to empty the buffer, but they are executed only if the event is set. While the event is not set, the channel is not executable.

When the buffer is empty, this algorithm restarts (a new buffer descriptor is read).

#### **A.3.2.2 ssiapp\_2\_mcu**

This script performs a data move from the 8, 16, 24 or 32-bit FIFO of the SSI to the AP buffer.

The SSI is in the AP region and 2 FIFO enabled. The MCU buffer is accessed through the burst DMA while the SSI is accessed through Per DMA.

##### **A.3.2.2.1 Parameters Transmitted Through the Context ssiapp\_2\_mcu**

r0: mask to check events2 - If script is triggered by event 32+I, r0[I] must be set to 1. (Project dependent)

r1: mask to check event - If script is triggered by event I, r1[I] must be set to 1. (Project dependent)

r6: address of the SSI RX FIFO 0

r7: Watermark level - Used to determine the maximum of data that can be sent to the peripheral each time the channel is started.

#### **A.3.2.2.2 Parameters Transmitted Through the Buffer Descriptor ssiapp\_2\_mcu**

The data length (8, 16, 24, 32 bits) is set in the command field of the first buffer descriptor word, especially bits 24 and 25.

The number of bytes to transmit is stored in the first buffer descriptor word (count field)

The destination address is stored in the second buffer descriptor word (address field).

The Extended Buffer address is not used

#### **A.3.2.2.3 Use of the General Register During the Execution ssiapp\_2\_mcu**

r0 = mask to check events2

r1 = mask to check events

r2 = AP M3 destination address

r3 = Ram channel context address

r4 = Bd mode

r5 = Bd mode Count

r6 = SDMA memory mapped regs base address

r7 = Watermark

#### **A.3.2.2.4 Overview of Script Functionality ssiapp\_2\_mcu**

When the SSI sends an event, a data transfer loop is executed. This event is set when there is at least "watermark level" of data in the FIFO(sum of left and right FIFOs).

During this data transfer loop, the number of bytes that are transferred from the RX FIFO of the SSI to the EMI is equal to the minimum value between the "watermark level" and the number of data that have not been transferred yet. For each data transfer (one read, one write), the size of the memory write access is fixed by the parameter data length.

The necessary numbers of transfer loops are executed in order to fill the buffer, but they are executed only if the event is set. While the event is not set, the channel is not executable.

When the buffer is full, this algorithm restarts (a new buffer descriptor is read).

### A.3.2.3 mcu\_2\_ssish

This script is similar to mcu\_2\_ssiapp.

The only difference is the way SDMA core accesses the SSI TX FIFO. For script of mcu\_2\_ssiapp, SDMA core accesses the TX FIFO through Functional Unit Bus and regards it as the peripheral DMA, whereas for the mcu\_2\_ssish, the access is through the Shared Peripheral Bus and SSI can be directly accessed by SDMA.

### A.3.2.4 ssish\_2\_mcu

This script is similar to ssiapp\_2\_mcu. The only difference is the way SDMA core accesses the SSI RX FIFO.

For script of ssiapp\_2\_mcu, SDMA core accesses the RX FIFO through Functional Unit Bus and regards it as the peripheral DMA, whereas for the ssish\_2\_mcu, the access is through the Shared Peripheral Bus and SSI can be directly accessed by SDMA.

### A.3.2.5 p\_2\_p

This script performs a data move of 32 bits from the peripheral's FIFO connected either on SPBA or AIPS bus to another.

If destination FIFO belongs to ASRC then pad adding is performed after every N samples from the source device. If the source FIFO belongs to ASRC then after transmitting N samples a pad is swallowed. In case there is a transaction between ASRC(source) and SPDIF (destination) no pad adding or swallowing is required because SPDIF expects even number of samples.

This script presently does not deal with the transactions involving two devices connected to AIPS bus.

#### A.3.2.5.1 Parameters Transmitted Through the Context p\_2\_p

r0: LWML event mask

r1: HWML event mask

r2: source address

r6: destination address

r7: INFO. See details in the following table.

## Scripts

p\_2\_p (r7 INFO)

Bits	Name	Description
0-7	Lower WML	Watermark Level
8	PS	1 : Pad Swallowing 0 : No Pad swallowing
9	PA	1 : Pad Adding 0 : No Pad swallowing
10	SPDIF	If this bit is set both source and destination are on SPBA
11	Source Bit(SP)	1 : Source on SPBA 0 : Source on AIPS
12	Destination Bit (DP)	1 : Destination on SPBA 0 : Destination on AIPS
13-15	-----	MUST BE 0
16-23	Higher WML	HWML
24-27	N	Total number of samples after which Pad adding/Swallowing must be done. It must be odd.
28	Lower WML Event (LWE)	SDMA events reg to check for LWML event mask 0 : LWE in EVENTS register 1 : LWE in EVENTS2 register
29	Higher WML Event(HWE)	SDMA events reg to check for HWML event mask 0 : HWE in EVENTS register 1 : HWE in EVENTS2 register
30	-----	MUST BE 0
31	CONT	1 : Amount of samples to be transferred is unknown and script will keep on transferring samples as long as both events are detected and script must be manually stopped by the application 0 : The amount of samples to be is equal to the count field of mode word

### A.3.2.5.2 Parameters Transmitted Through the Buffer Descriptor p\_2\_p

In the first buffer descriptor word, the mode and number of types to transfer is included. The Buffer address and Extended Buffer address are not used.

### A.3.2.5.3 Use of the General Register During the Execution p\_2\_p

The register usage for each case(spba\_2\_spba, asrc\_2\_spba...) are different, so don't present here.



### A.3.2.5.4 Overview of Script Functionality p\_2\_p

The script get the number of data to transfer per event according to the bit 31(CONT) of INFO.

If CONT is 0, each transfer loop will transmit the number of data as in the mode count; Or else, it means the number of samples to be transferred is not known and the transfer must take place as long as both events are detected and the script must be manually stopped by the application in case the transfer needs to be stopped.

When PS or PA is set, then after each N words transfer, a pad swallowing or pad adding will be done. This is to handle the ASRC involved transfer.

The transfer is executed only when the event is set. After each transfer for the event, the script will try to obtain a new BD.

### A.3.2.6 hdmi\_dma

This script sets up the HDMI DMA start and end address from the buffer chain, and launches an HDMI DMA transfer on the HDMI DMA done event.

#### A.3.2.6.1 Parameters Transmitted Through the Context hdmi\_dma

r4: Number of DMA buffers in loop

r6: Address of data structure setup by software. Detailed data structure below:

**Table A-2. SDMA Scripts Overview**

Name	Offset	Width (bytes)	Description
CONTROL	0	4	Address of HDMI DMA control register
STATUS	4	4	Address of HDMI DMA status register
START	8	4	Address of HDMI DMA start address register
START1	12	4	Start address of buffer 1 for HDMI DMA transfer
END1	16	4	End address of buffer 1 for HDMI DMA transfer
START2	20	4	Start address of buffer 1 for HDMI DMA transfer. Available only when value of r4>1
END2	24	4	End address of buffer 1 for HDMI DMA transfer. Available only when value of r4>1

*Table continues on the next page...*

**Table A-2. SDMA Scripts Overview (continued)**

Name	Offset	Width (bytes)	Description
STARTn	4+8*n	4	Start address of buffer 1 for HDMI DMA transfer. n is the value of r4.
ENDn	8+8*n	4	End address of buffer 1 for HDMI DMA transfer. n is the value in r4.

#### **A.3.2.6.2 Parameters Transmitted Through the Buffer Descriptor `hdmi_dma`**

None

#### **A.3.2.6.3 Use of the General Register During the Execution `hdmi_dma`**

r0 = N/A

r1 = control register address

r2 = status register address

r3 = DMA start address register address

r4 = offset of DMA buffer chain end

r5 = offset of current DMA buffer

r6 = pointer of data structure that described above

r7 = N/A

#### **A.3.2.6.4 Overview of Script Functionality `hdmi_dma`**

The parameters of the data structure used in the script are retrieved from the channel context. No parameter is retrieved through the buffer descriptor as no real SDMA transfer performed.

When the HDMI DMA done request is captured, the script will clear the DMA done bit in the status register and set up the DMA start and end address to the next buffer (rewind if it reaches the end of the buffer chain). Next it will restart the HDMI DMA transfer, then yield and wait for the next DMA done request.

When an error is captured, it will stop the channel.



## **Appendix B**

### **i.MX 6SoloLite Revision History**

#### **B.1 Substantive changes from revision 0 to revision 1**

Substantive changes from revision 0 to revision 1 are as follows:

## B.1.1 Reference Manual Revision History

Reference	Description
-	<p>In the following chapters, signal names have been updated such that each name should uniquely identify a signal across all instances of all IP blocks.</p> <ul style="list-style-type: none"> <li>• ARM</li> <li>• AUDMUX</li> <li>• CCM</li> <li>• CSI</li> <li>• ECSPI</li> <li>• EIM</li> <li>• EPDC</li> <li>• ESAI</li> <li>• EPIT</li> <li>• FEC</li> <li>• GPT</li> <li>• I2C</li> <li>• KPP</li> <li>• LCDIF</li> <li>• MMDC</li> <li>• PWM</li> <li>• SDMA</li> <li>• SJC</li> <li>• SNVS</li> <li>• SPDC</li> <li>• SPDIF</li> <li>• SRC</li> <li>• UART</li> <li>• USDHC</li> <li>• WDOG</li> <li>• XTALOSC</li> </ul> <p>There is an ongoing effort to align all documentation to the standardized signal names presented in <i>i.MX 6 Series Signal Name Mapping (EB792)</i> on Compass Extranet.</p>
-	Module clock topics have been added/updated to align each chapter with CCM.
<a href="#">Secure Non-Volatile Storage (SNVS)</a>	SNVS chapter added to the Reference Manual.

## B.1.2 Architecture Overview Revision History

Reference	Description
<a href="#">Features</a>	New version of Multimedia Card System Specification added to features list.

### B.1.3 Memory Maps Revision History

Reference	Description
-	Updated addresses in the memory map

### B.1.4 Interrupts Revision History

No substantive changes

### B.1.5 External Signals Revision History

Reference	Description
<a href="#">Muxing Options</a> <a href="#">Pin Assignments</a>	<ul style="list-style-type: none"> <li>- Changed EIM_EBx to EIM_EBx_B.</li> <li>- Changed EIM_OE to EIM_OE_B.</li> <li>- Changed EIM_CSx to EIM_CSx_B.</li> <li>- Changed EIM_LBA to EIM_LBA_B.</li> <li>- Changed EIM_WAIT to EIM_WAIT_B.</li> <li>- Changed DRAM_CAS to DRAM_CAS_B.</li> <li>- Changed DRAM_RAS to DRAM_RAS_B.</li> <li>- Changed DRAM_CSx to DRAM_CSx_B.</li> <li>- Changed DRAM_SDWE to DRAM_SDWE_B.</li> </ul>
<a href="#">Pin Assignments</a>	- Changed padField ODT value = 5 from RESERVED0 to 24_OHM.

### B.1.6 Fusemap Revision History

Reference	Description
<a href="#">Boot Fusemap</a>	<ul style="list-style-type: none"> <li>• Throughout: Replaced "CS select (SPI only)" with "eCSPI chip select."</li> <li>• Throughout: Replaced "CS#n" with "ECSPiX_SSn"</li> <li>• Throughout: Replaced "SPI" with "eCSPI."</li> <li>• Throughout: Replaced "eSDHC" with "uSDHC."</li> </ul>
<a href="#">Fusemap Descriptions Table</a>	Added new rows to "Fusemap Descriptions" table.

### B.1.7 External Memory Controllers Revision History

No substantive changes

## B.1.8 System Debug Revision History

No substantive changes

## B.1.9 System Boot Revision History

Reference	Description
<a href="#">IOMUX Configuration for SD/MMC</a>	Added row for the Card Detect signal.
<a href="#">Serial ROM eFUSE Configuration</a>	Replaced "CS#n" with "ECSPix_SSn".
<a href="#">Write Data Command</a>	Added EIM - Register address range to "Valid DCD Address Ranges" table.

## B.1.10 Multimedia Revision History

No substantive changes

## B.1.11 Power Management Revision History

Reference	Description
-	Added information on the Display domain for i.MX 6SoloLite.
<a href="#">Crystal Oscillator (XTALOSC)</a>	Added description of RC oscillator.

## B.1.12 System Security Revision History

Reference	Description
<a href="#">Overview</a>	Added two bullets to Overview: "Data co-processor (DCP) with SHA-1/SHA-256 hashing algorithms, and AES encryption algorithm with device unique secret key for secure off-chip storage" and "Random number generator (RNGB)".
<a href="#">Overview</a>	Added SL system figure
<a href="#">Overview</a>	Added SJC and HAB overviews

## B.1.13 ARM A9 Revision History

No substantive changes

### B.1.14 AIPSTZ Revision History

No substantive changes

### B.1.15 AUDMUX Revision History

No substantive changes

### B.1.16 CCM Revision History

Reference	Description
<a href="#">CCM Memory Map/Register Definition</a>	Filled in empty bitfield descriptions for reserved bitfields.
<a href="#">CCM Memory Map/Register Definition</a>	Updated ccm_analog_pll_sys_ss field.
<a href="#">CCM Memory Map/Register Definition</a>	Added text to REG_BYPASS_COUNT bitfield.
<a href="#">CCM Clock Tree</a>	Frequency dividers added to Audio and Video PLLs in CCM Clock Tree.
<a href="#">Clock Root Generator</a>	Updated ACLK_CLK_ROOT branch figure.
<a href="#">Clock Switcher</a>	Switcher clock figure updated. Dividers added to Audio and Video PLLs.
<a href="#">Phase Fractional Dividers (PFD)</a>	Topic updated, reference to app note EB790 added.
<a href="#">CCM Memory Map/Register Definition</a>	CGPR, CSCMR1, CCSR, CLPCR, and CCGR1 registers have been updated.
<a href="#">CCM Memory Map/Register Definition</a>	CBCDR and CBCMR reset values updated.

### B.1.17 CSI Revision History

No substantive changes

### B.1.18 DBGMON Revision History

No substantive changes

### B.1.19 DCP Revision History

No substantive changes

## **B.1.20 ECSPi Revision History**

No substantive changes

## **B.1.21 EIM Revision History**

No substantive changes

## **B.1.22 eLCDIF Revision History**

No substantive changes

## **B.1.23 EPDC Revision History**

No substantive changes

## **B.1.24 EPIT Revision History**

No substantive changes

## **B.1.25 FEC Revision History**

No substantive changes

## **B.1.26 GPC Revision History**

No substantive changes

## **B.1.27 GPIO Revision History**

No substantive changes



## B.1.28 GPT Revision History

No substantive changes

## B.1.29 GPU2D Revision History

No substantive changes

## B.1.30 I2C Revision History

No substantive changes

## B.1.31 IOMUXC Revision History

Reference	Description
<a href="#">IOMUXC</a>	Removed instance and in_pin information from nodes in DAISY field descriptions. Clarified the UART RTS and RX_DATA input select descriptions. Corrected the description of the DAISY field.
<a href="#">IOMUXC</a>	<ul style="list-style-type: none"> <li>- Changed EIM_EBx to EIM_EBx_B.</li> <li>- Changed EIM_OE to EIM_OE_B.</li> <li>- Changed EIM_CSx to EIM_CSx_B.</li> <li>- Changed EIM_LBA to EIM_LBA_B.</li> <li>- Changed EIM_WAIT to EIM_WAIT_B.</li> <li>- Changed DRAM_CAS to DRAM_CAS_B.</li> <li>- Changed DRAM_RAS to DRAM_RAS_B.</li> <li>- Changed DRAM_CSx to DRAM_CSx_B.</li> <li>- Changed DRAM_SDWE to DRAM_SDWE_B.</li> </ul>
<a href="#">IOMUXC</a>	- Changed padField ODT value = 5 from RESERVED0 to 24_OHM.

## B.1.32 KPP Revision History

No substantive changes

## B.1.33 MMDC Revision History

Reference	Description
<a href="#">ZQ calibration</a>	ZQ calibration description updated.

## B.1.34 NIC Revision History

No substantive changes

## B.1.35 OCOTP Revision History

No substantive changes

## B.1.36 OCRAM Revision History

No substantive changes

## B.1.37 PMU Revision History

Reference	Description
<a href="#">Overview</a>	Removed switch from ARM L2 cache in "Typical Power Diagram" figure.

## B.1.38 PWM Revision History

No substantive changes

## B.1.39 PXP Revision History

No substantive changes

## B.1.40 RNGB Revision History

Reference	Description
<a href="#">RNG</a>	Added ER, VCR, XKEY, OCCR, OSC_CNT registers.
<a href="#">Deterministic Mode</a> <a href="#">Verification Mode</a> <a href="#">Oscillator Frequency Verification</a>	Added "Deterministic Mode", "Verification Mode", and "Oscillator Frequency Verification" topics.

## B.1.41 ROMCP Revision History

No substantive changes

## B.1.42 SDMA Revision History

No substantive changes

## B.1.43 SJC Revision History

No substantive changes

## B.1.44 SNVS Revision History

No substantive changes

## B.1.45 SPBA Revision History

No substantive changes

## B.1.46 SPDC Revision History

Reference	Description
<a href="#">External Signals</a>	Updated "I/O" setting of SPDC_XDIOR, SPDC_XDIOL, SPDC_YDIODL, and SPDC_YDIOUL to "O".

## B.1.47 SPDIF Revision History

No substantive changes

## B.1.48 SRC Revision History

Reference	Description
<a href="#">Reset and Power-up Flow</a>	Chip reset scheme figures are updated.
<a href="#">Power mode transitions</a>	Power modes transition table is updated.
<a href="#">Reset inputs and outputs</a>	Reset inputs and outputs topic updated. SRC reset table added.

## B.1.49 SSI Revision History

No substantive changes

## B.1.50 TEMPMON Revision History

No substantive changes

## B.1.51 TZASC Revision History

No substantive changes

## B.1.52 UART Revision History

Reference	Description
<a href="#">CTS - Clear To Send</a> <a href="#">RTS - Request To Send</a>	Updated input and output modes.

## B.1.53 USB Revision History

Reference	Description
<a href="#">Pins Used for OTG 1 Controller</a>	Added two additional pin descriptions.
<a href="#">Split Transaction Isochronous Transfer Descriptor (siTD)</a>	Added footnotes to indicate static endpoint state and transfer results ranges.
<a href="#">Queue Element Transfer Descriptor (qTD)</a>	Added footnotes to indicate transfer results range. range.
<a href="#">Queue Head</a>	Added footnotes to indicate static endpoint state, transfer overlay, and transfer results ranges.  Added footnote to several fields to indicate they are used exclusively to support split transactions to USB 2.0 hubs.
<a href="#">Periodic Frame Span Traversal Node (FSTN)</a>	Added footnote to Typ field in third row: "Must be set to indicate a queue head."
<a href="#">Endpoint Queue Head (dQH)</a>	Updated field delineations in "Endpoint Queue Head (dQH)" table. Added footnote to indicate transfer overlay range.

## B.1.54 USB PHY Revision History

No substantive changes

## B.1.55 USDHC Revision History

No substantive changes

## B.1.56 WDOG Revision History

No substantive changes

## B.1.57 XTALOSC Revision History

No substantive changes

## B.1.58 SDMA Scripts Revision History

Reference	Description
<a href="#">Overview of Script Functionality uartsh_2_mcu</a>	Added figure.



### ***How to Reach Us:***

#### **Home Page:**

freescale.com

#### **Web Support:**

freescale.com/support

Information in this document is provided solely to enable system and software implementers to use Freescale products. There are no express or implied copyright licenses granted hereunder to design or fabricate any integrated circuits based on the information in this document.

Freescale reserves the right to make changes without further notice to any products herein. Freescale makes no warranty, representation, or guarantee regarding the suitability of its products for any particular purpose, nor does Freescale assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters that may be provided in Freescale data sheets and/or specifications can and do vary in different applications, and actual performance may vary over time. All operating parameters, including "typicals," must be validated for each customer application by customer's technical experts. Freescale does not convey any license under its patent rights nor the rights of others. Freescale sells products pursuant to standard terms and conditions of sale, which can be found at the following address: [freescale.com/SalesTermsandConditions](http://freescale.com/SalesTermsandConditions).

Freescale and the Freescale logo are trademarks of Freescale Semiconductor, Inc., Reg. U.S. Pat. & Tm. Off. ARM and Cortex are registered trademarks of ARM Limited. ARM Cortex-A9 is the trademark of ARM Limited. Synopsys portions Copyright © 2013 Synopsys, Inc. Used with permission. Synopsys & DesignWare are registered trademarks of Synopsys, Inc. All other product or service names are the property of their respective owners.

© 2013 Freescale Semiconductor, Inc.

